



UNIVERSIDAD VERACRUZANA

BIBLIOTECA SERVICE

14



MAYO



2021

**TECNOLOGÍAS PARA LA INTEGRACIÓN DE
SOLUCIONES**

ROJANO CACERES JOSE RAFAEL

EQUIPO 5

ALUMNOS:

- VASQUEZ RENDON JAIR
- ZARATE RODRIGUEZ ANDRES
- HERNANDEZ CABRERA HORUS ALEJANDRO
- GRAJALES MANCILLA LITZY ARISBETH

Tabla de contenido

<i>Introducción</i>	<i>1</i>
<i>Motivación</i>	<i>1</i>
<i>Problemática</i>	<i>1</i>
<i>Solución</i>	<i>1</i>
<i>Costos</i>	<i>2</i>
<i>Diagrama de despliegue</i>	<i>3</i>
<i>Documentación del API SOAP y REST:</i>	<i>4</i>
EndPoint	4
Parámetros de recepción	7
GuardarLibroRequest:	7
BuscarLibroRequest:	7
MostrarLibrosRequest:	7
ModificarLibroRequest:	7
EliminarLibroRequest:	8
TotalLibroRequest:	8
Parámetros devueltos	9
GuardarLibroResponse:	9
BuscarLibroResponse:	9
MostrarLibrosResponse:	9
ModificarLibroResponse:	10
EliminarLibroResponse:	10
TotalLibroResponse:	10
Plan de pruebas, se puede realizar siguiendo ejercicios con curl	11
Dockerfile del microservicio-libro	16
Script ejecutar.sh	16
<i>Proyecto publicado en github</i>	<i>16</i>
<i>Proyecto desplegado</i>	<i>16</i>

Introducción

El presente proyecto tiene como objetivo evaluar y aplicar los conocimientos adquiridos durante el curso de Tecnologías para la integración de soluciones, para ello se tiene planeado realizar un servicio de API'S enfocado a una biblioteca, en el cual emplearemos las tecnologías SOAP y REST para su desarrollo, los lineamientos de entrega nos indican que solo trabajaremos sobre el backend.

De forma que la parte visual en esta ocasión estará ausente, para brindar mayor información sobre cómo hacer uso de los microservicios de nuestro web service, indicaremos más adelante cuales son los parámetros de recepción requeridos para utilizar el microservicio y cuáles serán los parámetros de devolución.

Motivación

Nuestra motivación es ayudar con la facilitación y el buen manejo administrativo de las bibliotecas, dado que muchas de ellas aún se siguen administrando de una forma tradicional realizando sus registros en papel o bitácoras como son mayormente conocidas, para ello, se plantea desarrollar un web service que contribuya con esta causa, puesto que les permitirá realizar la gran mayoría de las actividades en un menor tiempo.

Problemática

Se crea a raíz de que actualmente algunas bibliotecas aún llevan el control a lápiz y papel, además de que en la mayoría de los casos al momento de registrarse en la biblioteca y tratar de solicitar el préstamo de un libro los formularios que suelen utilizar son muy extensos y complicados de realizar, lo que complica gran parte de las actividades dentro de la biblioteca como pueden ser la consulta y renovación de libros.

Solución

Para la solución de esta problemática se planteó un servicio de API el cual está dividido en microservicios siendo estos:

- Servicio de libros : incorpora un registro de libros, donde, se podrá introducir los datos relacionados a él. Para poder realizar esta acción se tendrá que introducir el título del libro, autor, editorial, categoría, descripción y estatus del préstamo, para poder guardarlo en la base de datos. También, se tomó en cuenta poder eliminar los libros que se desee, para ello, solo se requiere que se introduzca el id del libro, para que esta acción se realice.

De igual manera, se tiene la opción de modificar los libros, para eso, solo se necesita insertar los datos a cambiar. Otra acción que se puede realizar es el de buscar libros, ya que, solo se requiere insertar el id y título del libro, con el fin de reducir el tiempo de la búsqueda de un libro en específico.

Las últimas acciones que se le incorporó al microservicio es el de mostrar libros, dado que, muestra todos los libros que se encuentran registrados en la base de datos, por último, es la de total de libros, donde se contabiliza la cantidad de libros registrados.

- Servicio de préstamos : incorpora el siguiente préstamo del libro, renovación de préstamo, status de préstamos y un historial de préstamos.
- Servicio público: incorpora un registro del usuario que contará con los siguientes atributos: un Id, Nombre, Correo, Domicilio y Prioridad.

Logrando recuperar el usuario por id , realizar una modificación y clasificar al usuario por prioridad.

Costos

A continuación indicaremos las diversos medios que se utilizaron para el desarrollo de los microservicios:

- Para el despliegue de los microservicios utilizamos Heroku en su versión gratuita, lo cual no generó gastos adicionales.



- Para el hosting del servicio de base de datos utilizamos una versión de prueba de Clever-Cloud, la cual tampoco generó gastos.



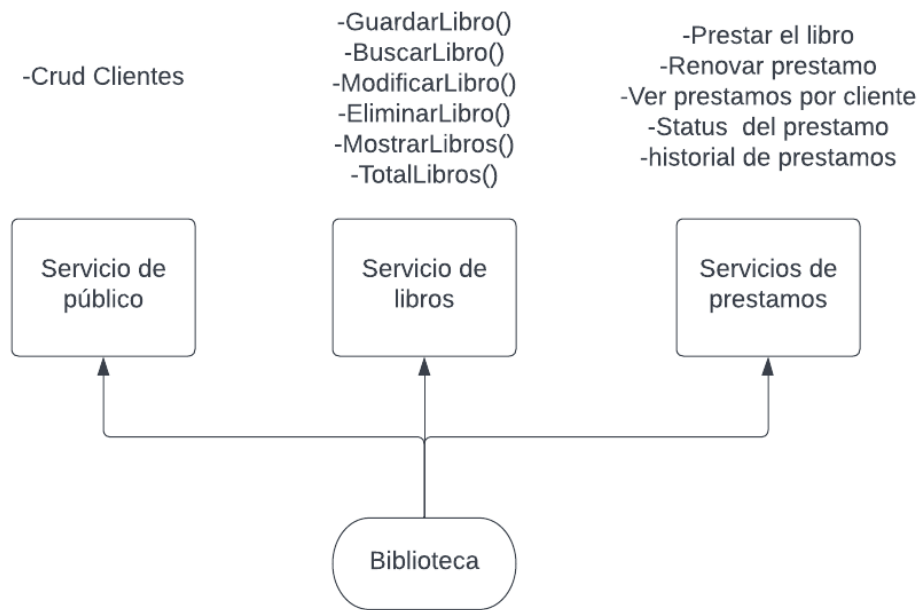
- En cuanto a la mano de obra no fue necesario el pago de sueldos ni la contratación de nuevo personal, puesto que el desarrollo está siendo realizado por el equipo de trabajo.



- No fue necesario adquirir recursos extra para el desarrollo de este proyecto, dado que utilizamos los recursos que ya teníamos a nuestra disposición como es el caso de un ordenador en el cual programar, servicio de internet, etc.



Diagrama de despliegue



Documentación del API SOAP y REST:

EndPoint

Microservicio Libro

Este método hace referencia a 2 elementos de nuestro archivo wsdl, los cuales son GuardarLibroRequest y GuardarLibroResponse, de forma general este método recupera los datos que son ingresados en el wsdl, genera un objeto de tipo libro el cual posteriormente es guardado en una base de datos, por último se envía un mensaje de respuesta indicando que el libro se agregó correctamente.

```
//El todo Guardar Libro "Hace referencia a los elementos GuardarLibroRequest y GuardarLibroResponse del archivo Libro.xsd"
@PayloadRoot(localPart = "GuardarLibroRequest", namespace = "https://t4is.uv.mx/libro")
@ResponsePayload
public GuardarLibroResponse guardarLibro(@RequestPayload GuardarLibroRequest petition) { //El parámetro petition nos ayuda a recuperar valores ingresados en el wsdl
    GuardarLibroResponse respuesta = new GuardarLibroResponse();
    //Se crea un nuevo objeto de tipo Libro
    Libro libro = new Libro();
    //Pedir los siguientes datos que serán almacenados en la base de datos:

    //Nombre del título (tipo de dato String)
    libro.setTitulo(petition.getTitulo());
    //Nombre del Autor (tipo de dato String)
    libro.setAutor(petition.getAutor());
    //Nombre de la editorial (tipo de dato String)
    libro.setEditorial(petition.getEditorial());
    //Categoría a la cual pertenecerá (tipo de dato String)
    libro.setCategoria(petition.getCategoria());
    //Una breve descripción del libro (tipo de dato String)
    libro.setDescripcion(petition.getDescripcion());
    //El estatus del libro el cual reflejara si está disponible el libro o no (tipo de dato String)
    libro.setStatusPrestamo(petition.getStatusPrestamo());
    //Por último guarda la información del libro en la base de datos
    libro.save(libro);
    //Y manda una respuesta de "El libro T4IS se agregó correctamente"
    respuesta.setRespuesta("El libro "+petition.getTitulo() + " se agregó correctamente");
    return respuesta;
}
```

Este método hace referencia a 2 elementos de nuestro archivo wsdl, los cuales son BuscarLibroRequest y BuscarLibroResponse, de forma general este método recupera los libros ya almacenados en la base de datos, según lo solicitado como búsqueda, por último los guarda regresando como respuesta al wsdl.

```

// TODO: buscar libro
@PayloadRoot(localPart = "BuscarLibroRequest", namespace = "https://t4is.uv.mx/libro")
@ResponsePayload
public BuscarLibroResponse buscarLibro(@RequestPayload BuscarLibroRequest peticion) { //El parametro peticion nos ayuda a recuperar valores ingresados en el wsdl
    BuscarLibroResponse respuesta = new BuscarLibroResponse();
    //Pide el ID del libro del cual se requiere la informacion
    String id = Integer.toString(peticion.getId());
    //Se utiliza un arreglo el cual buscara Los Libros utilizando su ID y su Titulo (Datos tipo Strings)
    Iterable<Libro> listaLibros = iLibro.findByIdAndTitulo(peticion.getId(), peticion.getTitulo());
    //Se realiza una busqueda entre todos Los datos de La base de datos
    for (Libro libro : listaLibros){
        //Devuelve el libro solicitado si se encuentran Los datos solicitados en La base de datos
        BuscarLibroResponse.Libros l = new BuscarLibroResponse.Libros();
        //Devuelve el atributo ID (Tipo de dato int)
        l.setId(libro.getId());
        //Devuelve el nombre del titulo (Tipo de dato String)
        l.setTitulo(libro.getTitulo());
        //Devuelve el nombre del autor (Tipo de dato String)
        l.setAutor(libro.getAutor());
        //Devuelve el nombre de La editorial (Tipo de dato String)
        l.setEditorial(libro.getEditorial());
        //Devuelve el nombre de La categoria (Tipo de dato String)
        l.setCategoria(libro.getCategoria());
        //Devuelve La descripcion del libro (Tipo de dato String)
        l.setDescripcion(libro.getDescripcion());
        //Devuelve el estado del libro (Tipo de dato String)
        l.setStatusPrestamo(libro.getStatusPrestamo());
        respuesta.getLibros().add(l);
    }
    return respuesta;
}

```

Este método hace referencia a 2 elementos de nuestro archivo wsdl, los cuales son MostrarLibrosRequest y MostrarLibrosResponse, de forma general este método recupera todos los libros que ya están almacenados en la base de datos, posteriormente los guarda en un arreglo, el cual es enviado como respuesta al wsdl.

```

// TODO: Mostrar Libros "Hac referencia a Los elementos MostrarLibrosRequest y MostrarLibrosResponse del archivo Libro.xsd"
@PayloadRoot(localPart = "MostrarLibrosRequest", namespace = "https://t4is.uv.mx/libro")
@ResponsePayload
public MostrarLibrosResponse mostrarLibros(){
    MostrarLibrosResponse respuesta = new MostrarLibrosResponse();
    //Se crea un objeto iterable del objeto Libro, el cual permitira recuperar todos Los libros almacenados en La base de datos
    Iterable<Libro> lista = iLibro.findAll();

    //Por cada objeto libro que sea encontrado en La base de datos, recuperar sus valores y Los guardar en una lista para mandarLos posteriormente
    for (Libro libro : lista) {
        MostrarLibrosResponse.Libros l = new MostrarLibrosResponse.Libros();
        l.setId(libro.getId());
        l.setTitulo(libro.getTitulo());
        l.setAutor(libro.getAutor());
        l.setEditorial(libro.getEditorial());
        l.setCategoria(libro.getCategoria());
        l.setDescripcion(libro.getDescripcion());
        l.setStatusPrestamo(libro.getStatusPrestamo());
        respuesta.getLibros().add(l);
    }
    //El valor que se regresa como respuesta es una lista con todos Los datos de Los libros que fueron recuperados de La base datos
    return respuesta;
}

```

Este método hace referencia a 2 elementos de nuestro archivo wsdl, los cuales son ModificarLibrosRequest y ModificarLibrosResponse, de forma general este método recupera todos los libros que ya están almacenados en la base de datos posteriormente regresa como res.


```

//Método Modificar Libro "Hace referencia a Los elementos ModificarLibroRequest y ModificarLibroResponse del archivo Libro.xsd"
@PayloadRoot(localPart = "ModificarLibroRequest", namespace = "https://t4is.uv.mx/libro")
@ResponsePayload
public ModificarLibroResponse modificarLibro(@RequestPayload ModificarLibroRequest petition){ //El parametro petition nos ayuda a recuperar valores ingresados en el wsdl
    ModificarLibroResponse respuesta = new ModificarLibroResponse();

    //Se instancia una objeto de la clase Libro
    Libro libro = new Libro();
    //Para este metodo el id es el valor que indica que libro de la base de datos se va a modificar
    libro.setId(petition.getId());
    //Los siguientes valores se reemplazaran los valores anteriores del libro
    libro.setTitulo(petition.getTitulo());
    //Solicita el nuevo nombre del Autor (Tipo de dato String));
    libro.setAutor(petition.getAutor());
    //Solicita el nuevo nombre de la Editorial (Tipo de dato String));
    libro.setEditorial(petition.getEditorial());
    //Solicita el nuevo nombre de la Categoría (Tipo de dato String));
    libro.setCategoria(petition.getCategoria());
    //Solicita el nuevo nombre de la Descripción (Tipo de dato String));
    libro.setDescripcion(petition.getDescripcion());
    //Solicita el nuevo Estatus del Libro (Tipo de dato String));
    libro.setStatusPrestamo(petition.getStatusPrestamo());
    //Guardamos los cambios en la base de datos
    ilibro.save(libro);
    //Enviamos un mensaje como respuesta indicando que el libro fue modificado
    respuesta.setRespuesta("Libro modificado");
    return respuesta;
}

```

Este método hace referencia a 2 elementos de nuestro archivo wsdl, los cuales son EliminarLibroRequest y EliminarLibroResponse, de forma general este método recupera los datos que son ingresados en el wsdl, elimina el libro del id ingresado, por último se envía un mensaje de respuesta indicando que el libro fue eliminado.

```

//Método Eliminar Libro "Hace referencia a Los elementos EliminarLibroRequest y EliminarLibroResponse del archivo Libro.xsd"
@PayloadRoot(localPart = "EliminarLibroRequest", namespace = "https://t4is.uv.mx/libro")
@ResponsePayload
public EliminarLibroResponse eliminarLibro(@RequestPayload EliminarLibroRequest petition){ //El parametro petition nos ayuda a recuperar valores ingresados en el wsdl
    EliminarLibroResponse respuesta = new EliminarLibroResponse();
    //Para este metodo recuperamos e indicamos el id del libro que vamos a eliminar
    ilibro.deleteById(petition.getId());
    //Enviamos un mensaje como respuesta indicando que el libro fue eliminado
    respuesta.setRespuesta("Libro eliminado");
    return respuesta;
}

```

Este método hace referencia a 2 elementos de nuestro archivo wsdl, los cuales son TotalLibroRequest y TotalLibroResponse, de forma general este método hace un conteo de todos los libros almacenados en la base de datos y regresa como respuesta el total de libros.

```

//Método Total Libros "Hace referencia a Los elementos TotalLibroRequest y TotalLibroResponse del archivo Libro.xsd"
@PayloadRoot(localPart = "TotalLibroRequest", namespace = "https://t4is.uv.mx/libro")
@ResponsePayload
public TotalLibroResponse totalLibros(){
    TotalLibroResponse respuesta = new TotalLibroResponse();
    //Este metodo lo que hace es realizar un conteo de todos los libros almacenado en la base de datos.
    respuesta.setRespuesta("Tu total de libros es: "+ilibro.count());
    //Se envia como respuesta el total de libros encontrados en la base de datos.
    return respuesta;
}

```

Microservicio Prestamo

```

//Realizar un prestamo hace referencia a los elementos GuardarPrestamoRequest y GuardarPrestamoResponse del archivo prestamo.xsd
@PayloadRoot(localPart = "GuardarPrestamoRequest", namespace = "https://t4is.uv.mx/prestamo")
@ResponsePayload
public GuardarPrestamoResponse guardarPrestamo(@RequestPayload GuardarPrestamoRequest petition) {
    GuardarPrestamoResponse respuesta = new GuardarPrestamoResponse();

    //Generamos el objeto prestamo recuperando los valores del archivo xsd, utilizando petition.get por cada uno de los datos.
    //Después insertamos el registro en la base de datos con la ayuda de la interfaz iprestamo y devolvemos como respuesta
    //un pequeño mensaje indicando el nombre del cliente el libro que apartó y la fecha hasta la que se realizó el prestamo.
    Prestamo prestamo = new Prestamo();

    prestamo.setFinicio(petition.getFinicio());
    prestamo.setFfin(petition.getFfin());
    prestamo.setNomcliente(petition.getNomcliente());
    prestamo.setTitulolibro(petition.getTitulolibro());
    prestamo.setStatus(petition.getStatus());
    prestamo.setResponsable(petition.getResponsable());
    iprestamo.save(prestamo);
    respuesta.setRespuesta("Hola " + petition.getNomcliente() + " acaba de apartar el libro " + petition.getTitulolibro() + " hasta " + petition.getFfin());
    return respuesta;
}

```

```

//Ver todos los prestamos hace referencia a los elementos MostrarPrestamoRequest y MostrarPrestamoResponse del archivo prestamo.xsd
@PayloadRoot(localPart = "MostrarPrestamoRequest", namespace = "https://t4is.uv.mx/prestamo")
@ResponsePayload
public MostrarPrestamoResponse mostrarprestamo() {

    MostrarPrestamoResponse respuesta = new MostrarPrestamoResponse();

    //Este método itera y recupera todos los prestamos almacenados en la base de datos.
    Iterable<Prestamo> lista = iprestamo.findAll();

    //Después de recuperar todos los prestamos generamos los objetos prestamos con cada uno de sus valores, los almacenamos
    //en la variable respuesta y los regresamos como respuesta en el microservicio.
    for (Prestamo o : lista) {
        MostrarPrestamoResponse.Prestamo e = new MostrarPrestamoResponse.Prestamo();
        e.setId(o.getId());
        e.setFinicio(o.getFinicio());
        e.setFfin(o.getFfin());
        e.setNomcliente(o.getNomcliente());
        e.setTitulolibro(o.getTitulolibro());
        e.setStatus(o.getStatus());
        e.setResponsable(o.getResponsable());
        respuesta.getPrestamo().add(e);
    }

    return respuesta;
}

```

```

//Renovar prestamo hace referencia a los elementos RenovarPrestamoRequest y RenovarPrestamoResponse del archivo prestamo.xsd
@PayloadRoot(localPart = "RenovarPrestamoRequest", namespace = "https://t4is.uv.mx/prestamo")
@ResponsePayload
public RenovarPrestamoResponse renovarprestamo (@RequestPayload RenovarPrestamoRequest petition){

    RenovarPrestamoResponse respuesta = new RenovarPrestamoResponse();

    //Esté metodo tiene como objetivo modificar la información de algun dato en el prestamo, como puede ser la fecha de inicio
    //y la fecha de fin, aunque también es posible realizar modificaciones de los demás parametros si así se requiere, el único
    //valor que no debe ser cambiado es el id, ya que este permite hacer match en la base de datos para localizar el registro. La
    //respuesta que recibimos es un mensaje indicando el titulo de libro y la nueva fecha hasta la que estará realizado el prestamo

    Prestamo prestamo = new Prestamo();
    prestamo.setId(petition.getId());
    prestamo.setFinicio(petition.getFinicio());
    prestamo.setFfin(petition.getFfin());
    prestamo.setNomcliente(petition.getNomcliente());
    prestamo.setTitulolibro(petition.getTitulolibro());
    prestamo.setStatus(petition.getStatus());
    prestamo.setResponsable(petition.getResponsable());
    iprestamo.save(prestamo);
    respuesta.setRespuesta("Ha renovado el prestamo del libro " + petition.getTitulolibro() + " hasta el " + petition.getFfin());

    return respuesta;
}

```

```
//Ver status del prestamo hace referencia a los elementos StatusPrestamoRequest y StatusPrestamoResponse del archivo prestamo.xsd
@PayloadRoot(localPart = "StatusPrestamoRequest", namespace = "https://t4is.uv.mx/prestamo")
@ResponsePayload
public StatusPrestamoResponse statusprestamo (@RequestPayload StatusPrestamoRequest petition){

    StatusPrestamoResponse respuesta = new StatusPrestamoResponse();

    //Este método itera y recupera todos los prestamos de la base de datos que correspondan al libro ingresado, se requieren de 2
    //parámetros los cuales son id y titulo del libro, estos valores son recuperados en petition.getId() y petition.getTituloLibro()
    //este método esta definido en la Interfaz IPrestamo
    Iterable<Prestamo> listaprestamos = iprestamo.findByIdAndTituloLibro(petition.getId(), petition.getTituloLibro());

    //Después de recuperar el prestamo ingresado, genera una respuesta en la que serán devueltos el titulo del libro y el estatus
    //del prestamo
    for (Prestamo prestamo : listaprestamos) {

        respuesta.setRespuesta("El status del libro: " + prestamo.getTituloLibro() + " es: " +prestamo.getStatus());

    }

    return respuesta;
}
```

```
//Buscar prestamo por cliente hace referencia a los elementos PrestamoPorClienteRequest y PrestamoPorClienteResponse del archivo prestamo.xsd
@PayloadRoot(localPart = "PrestamoPorClienteRequest", namespace = "https://t4is.uv.mx/prestamo")
@ResponsePayload
public PrestamoPorClienteResponse prestamocliente (@RequestPayload PrestamoPorClienteRequest petition){

    PrestamoPorClienteResponse respuesta = new PrestamoPorClienteResponse();

    //Este método itera y recupera todos los prestamos de la base de datos que correspondan al cliente ingresado en el campo nomCliente
    //y ese valor es recuperado en petition.getNomCliente(), este metodo esta definido en la Interfaz IPrestamo
    Iterable<Prestamo> listaPrestamosCliente = iprestamo.findByNomcliente(petition.getNomCliente());

    //Después de recuperar los prestamos del cliente, genera el objeto prestamo que será devuelto como response en el microservicio.
    for (Prestamo prestamo : listaPrestamosCliente) {
        PrestamoPorClienteResponse.Prestamo e = new PrestamoPorClienteResponse.Prestamo();
        e.setId(prestamo.getId());
        e.setFinicio(prestamo.getFinicio());
        e.setFfin(prestamo.getFfin());
        e.setNomcliente(prestamo.getNomcliente());
        e.setTituloLibro(prestamo.getTituloLibro());
        e.setStatus(prestamo.getStatus());
        e.setResponsable(prestamo.getResponsable());
        respuesta.getPrestamo().add(e);
    }
    return respuesta;
}
```

Microservicio Público

```
@Autowired
UsuarioS usuarioService;

@GetMapping()
public ArrayList<UsuarioM> obtenerUsuarios(){ //se regresa el arreglo de todos los usuarios de tipo ArrayList
    return usuarioService.obtenerUsuarios();
}
```

```
//regresa el usuarios pero actualizado
@PostMapping()
public UsuarioM guardarUsuario(@RequestBody UsuarioM usuario){ //tomar la informacion y guardarla en el objeto usuario
    return this.usuarioService.guardarUsuario(usuario); //llama al metodo del servicio pasandole el objeto que tiene la informacion
}
```

```
//https://microservicio-publico.herokuapp.com/usuario/2
@GetMapping( path =("/{id}") //buscar registro por su id
public Optional<UsuarioM> obtenerUsuarioPorId(@PathVariable("id") Long id) {
    return this.usuarioService.obtenerPorId(id);
}
```

```
//https://microservicio-publico.herokuapp.com/usuario/query?prioridad=2
@GetMapping("/query") //buscar por prioridad pasando los parametros por queryparams, poniendo la variable que queremos pasar y el valor
public ArrayList<UsuarioM> obtenerUsuarioPorPrioridad(@RequestParam("prioridad") Integer prioridad){
    return this.usuarioService.obtenerPorPrioridad(prioridad);
}
```

```
//https://microservicio-publico.herokuapp.com/usuario/2
@DeleteMapping( path =("/{id}") //nueva peticion http llamada delete , pasando el id
public String eliminarPorId(@PathVariable("id") Long id){
    boolean ok = this.usuarioService.eliminarUsuario(id);
    if (ok){
        return "Se eliminó el usuario con id " + id;
    }else{
        return "No pudo eliminar el usuario con id" + id;
    }
}
```

Parámetros de recepción

Servicio de libros

GuardarLibroRequest:

Para poder guardar un libro se requerirán los siguientes datos:

Título (dato String), Autor (dato String), Editorial (dato String), Categoría (dato String), Descripción (dato String) y Estatus del préstamo (dato String)

```
<GuardarLibroRequest xmlns="https://t4is.uv.mx/libro">
  <titulo>[string]</titulo>
  <autor>[string]</autor>
  <editorial>[string]</editorial>
  <categoria>[string]</categoria>
  <descripcion>[string]</descripcion>
  <statusPrestamo>[string]</statusPrestamo>
</GuardarLibroRequest>
```

BuscarLibroRequest:

Para poder realizar la búsqueda de un libro se requerirán los siguientes datos:

ID (dato int) y Título (dato String)

```
<BuscarLibroRequest xmlns="https://t4is.uv.mx/libro">
  <id>[int]</id>
  <titulo>[string]</titulo>
</BuscarLibroRequest>
```

MostrarLibrosRequest:

Para poder realizar la acción de mostrar todos los libros no es necesario ningún parámetro, puede ser ejecutado directamente.

```
<MostrarLibrosRequest xmlns="https://t4is.uv.mx/libro">[any]</MostrarLibrosRequest>
```

ModificarLibroRequest:

Para poder realizar una modificación a un libro existente se requerirán los siguientes datos:

ID (dato int), Título (dato String), Autor (dato String), Editorial (dato String), Categoria (dato String), Descripción (dato String) y Estatus del préstamo (dato String)

```
<ModificarLibroRequest xmlns="https://t4is.uv.mx/libro">
  <id>[int]</id>
  <titulo>[string]</titulo>
  <autor>[string]</autor>
  <editorial>[string]</editorial>
  <categoria>[string]</categoria>
  <descripcion>[string]</descripcion>
  <statusPrestamo>[string]</statusPrestamo>
</ModificarLibroRequest>
```

EliminarLibroRequest:

Para realizar la eliminación de un libro se requerirá el siguiente atributo:

ID (dato int)

```
<EliminarLibroRequest xmlns="https://t4is.uv.mx/libro">
  <id>[int]</id>
</EliminarLibroRequest>
```

TotalLibroRequest:

Para poder realizar la acción de ver el total de los libros no es necesario ningún parámetro, puede ser ejecutado directamente.

```
<TotalLibroRequest xmlns="https://t4is.uv.mx/libro">[any]</TotalLibroRequest>
```

Servicio de préstamos

Guardar préstamo

Para guardar un préstamo en la base de datos se solicita que se ingrese los datos que se piden como lo que es la fecha de inicio y fin, nombre del cliente, el título del libro a prestar, el estado y el nombre del responsable que prestó el libro.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <GuardarPrestamoRequest xmlns="https://t4is.uv.mx/prestamo">
      <finicio>3 de junio de 2022</finicio>
      <ffin>5 de junio de 2022</ffin>
      <nomcliente>Luis</nomcliente>
      <titulolibro>En las nubes</titulolibro>
      <status>Prestado</status>
      <responsable>Arisbeth</responsable>
    </GuardarPrestamoRequest>
  </Body>
</Envelope>
```

Mostrar Préstamo

Para saber cuáles son libros que se han prestado a los clientes , tenemos la opción de mostrarPrestamo, donde, le damos en Go y rápidamente nos mostrará una lista de los libros que han sido prestados independientemente del status.

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

Préstamo por cliente

Para saber si un cliente tiene libros de la biblioteca tenemos la opción de buscarlo, tan solo colocamos el nombre del cliente, le damos en Go y nos mostrará si ese cliente que introducimos tiene libros prestados de la biblioteca con todos los datos: fecha, título del libro, etc.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <PrestamoPorClienteRequest xmlns="https://t4is.uv.mx/prestamo">
      <nomCliente>Luis</nomCliente>
    </PrestamoPorClienteRequest>
  </Body>
</Envelope>
```

Renovar préstamo

Para renovar el préstamo de un libro solo tenemos que introducir los datos del préstamo anterior y renovarlo, tomando en cuenta que tienes que saber que id tiene ese préstamo para que se pueda hacer la renovación del préstamo del libro.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <RenovarPrestamoRequest xmlns="https://t4is.uv.mx/prestamo">
      <id>2</id>
      <finicio>3 de junio de 2022</finicio>
      <ffin>15 de junio de 2022</ffin>
      <nomcliente>Luis</nomcliente>
      <titulolibro>En las nubes</titulolibro>
      <status>Prestado</status>
      <responsable>Arisbeth</responsable>
    </RenovarPrestamoRequest>
  </Body>
</Envelope>
```

Status préstamo

Para saber el estado de un libro tenemos la opción del StatusPrestamo, para ello, solo tenemos que introducir el id y título del libro, automáticamente se encargará de mostrar si el libro está disponible o no.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <StatusPrestamoRequest xmlns="https://t4is.uv.mx/prestamo">
      <id>2</id>
      <titulolibro>En las nubes</titulolibro>
    </StatusPrestamoRequest>
  </Body>
</Envelope>
```

Servicio público

Obtener Usuarios

Para poder obtener los usuario que se tiene registrados en la base de datos, solo se requiere que coloquemos “usuario” en la url y que la consulta sea por el método GET. Automáticamente, nos devolverá los usuarios registrados.

GET



<https://microservicio-publico.herokuapp.com/usuario>

Guardar Usuario

Para poder guardar un usuario, solo colocamos “usuario” en la url, después, lo hacemos por el método POST y finalmente rellenamos los campos que se solicita del usuario.

POST



<https://microservicio-publico.herokuapp.com/usuario>

```
{
  .... "nombre": "m",
  .... "email": "m@.com",
  .... "prioridad": 2,
  .... "domicilio": "x",
  .... "telefono": 2
}
```

Modificar Usuario

Para poder realizar alguna modificación de un usuario, solo es cuestión de cambiar los campos que queremos modificar, finalmente lo hacemos por el método POST.

POST



<https://microservicio-publico.herokuapp.com/usuario>

```
{
  .... "id": 2,
  .... "nombre": "p",
  .... "email": "p@.com",
  .... "prioridad": 1,
  .... "domicilio": "p",
  .... "telefono": 1
}
```

Buscar Usuario por su id

Para buscar un usuario en especial, tenemos que poner “usuario/id” en la url, donde, el id es el número que tiene asignado el usuario. Con esto podremos buscar el usuario fácilmente, por ejemplo:

GET	▼	https://microservicio-publico.herokuapp.com/usuario/2
-----	---	---

OJO: tenemos que hacer la consulta por el método GET.

Buscar Usuario por su prioridad

Para poder realizar una búsqueda por prioridad se haría de la siguiente manera: colocamos en la url “usuario/query?prioridad=2”, donde, 2 es el número de prioridad del usuario.

Automáticamente mostrará el usuario que tenga esa prioridad. No olvidemos que esta petición se tiene que hacer por el método GET.

GET	▼	https://microservicio-publico.herokuapp.com/usuario/query?prioridad=2
-----	---	---

Eliminar Usuario por su id

Finalmente, tenemos el de eliminar usuario por su id, donde tenemos que poner en la url el id del usuario que se quiere eliminar. Para esta petición se tiene que utilizar el método de DELETE.

DELETE	▼	https://microservicio-publico.herokuapp.com/usuario/2
--------	---	---

Parámetros devueltos

Servicio de libros

GuardarLibroResponse:

La respuesta que recibimos después de guardar un libro es un mensaje indicando el título del libro y el estatus de la operación.

```
<ns2:GuardarLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">  
  <ns2:respuesta>El libro El señor de los anillos se agrego correctamente</ns2:respuesta>  
</ns2:GuardarLibroResponse>
```

BuscarLibroResponse:

La respuesta que recibimos después de buscar un libro, es el libro con sus respectivos datos.

```
<ns2:BuscarLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">  
  <ns2:libros>  
    <ns2:id>5</ns2:id>  
    <ns2:titulo>El señor de los anillos</ns2:titulo>  
    <ns2:autor>JR</ns2:autor>  
    <ns2:editorial>abcd123</ns2:editorial>  
    <ns2:categoria>Aventura, Fantasía</ns2:categoria>  
    <ns2:descripcion>Historia Medieval</ns2:descripcion>  
    <ns2:statusPrestamo>Reservado</ns2:statusPrestamo>  
  </ns2:libros>  
</ns2:BuscarLibroResponse>
```

MostrarLibrosResponse:

La respuesta que recibimos después de mostrar todos los libros, es el listado completo de libros almacenados en la base de datos.

```
<ns2:MostrarLibrosResponse xmlns:ns2="https://t4is.uv.mx/libro">  
  <ns2:libros>  
    <ns2:id>1</ns2:id>  
    <ns2:titulo>El código da vinci</ns2:titulo>  
    <ns2:autor>Da vinci</ns2:autor>  
    <ns2:editorial>DV</ns2:editorial>  
    <ns2:categoria>Historia</ns2:categoria>  
    <ns2:descripcion>Historia</ns2:descripcion>  
    <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>  
  </ns2:libros>
```

```
</ns2:MostrarLibrosResponse>
```

ModificarLibroResponse:

La respuesta que recibimos después de modificar un libro, es un mensaje indicando que el libro fue modificado.

```
<ns2:ModificarLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">  
  <ns2:respuesta>Libro modificado</ns2:respuesta>  
</ns2:ModificarLibroResponse>
```

EliminarLibroResponse:

La respuesta que recibimos después de eliminar un libro, es un mensaje indicando que el libro fue eliminado.

```
<ns2:EliminarLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">  
  <ns2:respuesta>Libro eliminado</ns2:respuesta>  
</ns2:EliminarLibroResponse>
```

TotalLibroResponse:

La respuesta que recibimos después de solicitar el total de libros, es un mensaje indicando el total de libros.

```
<ns2:TotalLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">  
  <ns2:respuesta>Tu total de libros es: 2</ns2:respuesta>  
</ns2:TotalLibroResponse>
```

Servicio de préstamos

Guardar préstamo

Cuando guardamos un préstamo automáticamente nos muestra un mensaje del nombre, el libro que fue prestado y la fecha que tiene el cliente para entregarlo.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">  
  <SOAP-ENV:Header/>  
  <SOAP-ENV:Body>  
    <ns2:GuardarPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">  
      <ns2:respuesta>Hola Luis acaba de apartar el libro En las nubes hasta 5 de junio de 2022</ns2:respuesta>  
    </ns2:GuardarPrestamoResponse>  
  </SOAP-ENV:Body>  
</SOAP-ENV:Envelope>
```

Mostrar préstamo

Cuando le damos en Go en MostrarPrestamo nos manda una lista de los libros que están o fueron prestado como lo podemos observar en la imagen.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:MostrarPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:prestamo>
        <ns2:id>1</ns2:id>
        <ns2:finicio>1 de junio 2022</ns2:finicio>
        <ns2:ffin>10 de junio 2022</ns2:ffin>
        <ns2:nomcliente>p</ns2:nomcliente>
        <ns2:titulolibro>libro1</ns2:titulolibro>
        <ns2:status>disponible</ns2:status>
        <ns2:responsable>m</ns2:responsable>
      </ns2:prestamo>
      <ns2:prestamo>
        <ns2:id>2</ns2:id>
        <ns2:finicio>3 de junio de 2022</ns2:finicio>
        <ns2:ffin>5 de junio de 2022</ns2:ffin>
        <ns2:nomcliente>Luis</ns2:nomcliente>
        <ns2:titulolibro>En las nubes</ns2:titulolibro>
        <ns2:status>Prestado</ns2:status>
        <ns2:responsable>Arisbeth</ns2:responsable>
      </ns2:prestamo>
    </ns2:MostrarPrestamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Préstamo por cliente

Como podemos observar cuando buscamos un préstamo por cliente, nos manda toda la información de ese préstamo en especial, con todos los datos registrados.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:PrestamoPorClienteResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:prestamo>
        <ns2:id>2</ns2:id>
        <ns2:finicio>3 de junio de 2022</ns2:finicio>
        <ns2:ffin>5 de junio de 2022</ns2:ffin>
        <ns2:nomcliente>Luis</ns2:nomcliente>
        <ns2:titulolibro>En las nubes</ns2:titulolibro>
        <ns2:status>Prestado</ns2:status>
        <ns2:responsable>Arisbeth</ns2:responsable>
      </ns2:prestamo>
    </ns2:PrestamoPorClienteResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Renovar préstamo

Como observamos cuando renovamos el préstamo de un libro nos manda como respuesta que el libro “En las nubes” será entregado hasta el día 15 de junio de 2022.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:RenovarPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:respuesta>Ha renovado el prestamo del libro En las nubes hasta el 15 de junio de 2022</ns2:respuesta>
    </ns2:RenovarPrestamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Status préstamo

Cuando solicitamos saber el status de un préstamo nos muestra como respuesta el dato de ese libro. Podemos verificar si ese libro se encuentra prestado o ya está disponible para los demás clientes.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:StatusPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:respuesta>El status del libro: En las nubes es: Prestado</ns2:respuesta>
    </ns2:StatusPrestamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Servicio público

Obtener usuario

El resultado de obtener usuario se mostrará de la siguiente manera:

```
[
  {
    "id": 2,
    "nombre": "j",
    "email": "j@.com",
    "prioridad": 2,
    "domicilio": "xalapa",
    "telefono": 22
  }
]
```

Guardar usuario

Cuando se guarda un usuario devuelve los campos que se insertó en el registro, por ejemplo:

```
{  
  "id": 3,  
  "nombre": "m",  
  "email": "m@.com",  
  "prioridad": 2,  
  "domicilio": "x",  
  "telefono": 2  
}
```

Modificar Usuario

Cuando se modifica un usuario muestra los campos con las modificaciones que se realizaron.

```
{  
  "id": 2,  
  "nombre": "p",  
  "email": "p@.com",  
  "prioridad": 1,  
  "domicilio": "p",  
  "telefono": 1  
}
```

Buscar Usuario por su id

Cuando se realiza una búsqueda devuelve el usuario con id que se colocó en la url.

```
{  
  "id": 2,  
  "nombre": "p",  
  "email": "p@.com",  
  "prioridad": 1,  
  "domicilio": "p",  
  "telefono": 1  
}
```

Buscar Usuario por su prioridad

Cuando se busca un usuario por prioridad muestra como resultado el usuario que tiene la prioridad que se colocó en la url, como el siguiente ejemplo:

```
[
  {
    "id": 3,
    "nombre": "m",
    "email": "m@.com",
    "prioridad": 2,
    "domicilio": "x",
    "telefono": 2
  }
]
```

Eliminar Usuario por su id

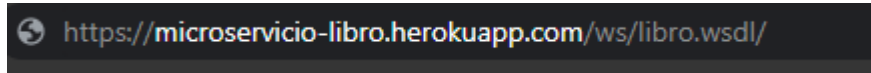
Cuando se elimina un usuario manda un mensaje como respuesta de que el usuario del id fue eliminado.

Se eliminó el usuario con id 2

Plan de pruebas

Servicio libro

Para acceder al wsdl requerimos de la siguiente línea: /ws/libro.wsdl/

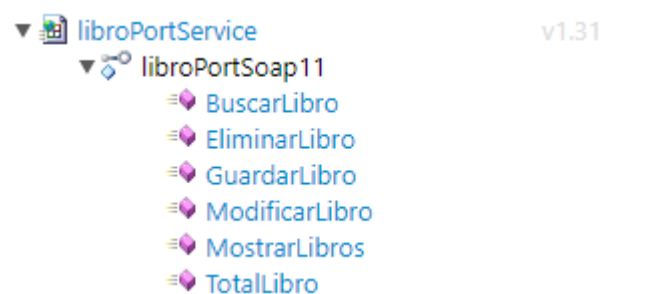


recuperando el wsdl:

```
This XML file does not appear to have any style information associated with it. The document tree is shown below.
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:sch="https://t4is.uv.mx/libro" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="https://t4is.uv.mx/libro" targetNamespace="https://t4is.uv.mx/libro">
  <wsdl:types>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" targetNamespace="https://t4is.uv.mx/libro">
      <!-- Guardar Libro -->
      <xs:element name="GuardarLibroRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="titulo" type="xs:string"/>
            <xs:element name="autor" type="xs:string"/>
            <xs:element name="editorial" type="xs:string"/>
            <xs:element name="categoria" type="xs:string"/>
            <xs:element name="descripcion" type="xs:string"/>
            <xs:element name="edadR" type="xs:int"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="GuardarLibroResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="respuesta" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <!-- Buscar libro -->
      <xs:element name="BuscarLibroRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="id" type="xs:int"/>
            <xs:element name="titulo" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="BuscarLibroResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element maxOccurs="unbounded" name="libros">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="id" type="xs:int"/>
                  <xs:element name="titulo" type="xs:string"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  </wsdl:types>
  <wsdl:binding name="GuardarLibro" type="tns:GuardarLibroRequest" style="rpc">
    <wsdl:operation name="GuardarLibro" input="tns:GuardarLibroRequest" output="tns:GuardarLibroResponse" style="rpc"/>
  </wsdl:binding>
  <wsdl:binding name="BuscarLibro" type="tns:BuscarLibroRequest" style="rpc">
    <wsdl:operation name="BuscarLibro" input="tns:BuscarLibroRequest" output="tns:BuscarLibroResponse" style="rpc"/>
  </wsdl:binding>

```

Mediante la herramienta de Wizzler que analiza los archivos WSDL y genera mensajes SOAP



Guardar Libro

Lo que se hará primero es guardar un nuevo libro, para ello, seleccionamos la opción de GuardarLibro, donde, se colocan los datos requeridos como se observa en la imagen, le damos en Go y listo el libro ha sido guardado.


```

POST https://microservicio-libro.herokuapp.com:443/ws
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <GuardarLibroRequest xmlns="https://t4is.uv.mx/libro">
      <titulo>Elon Musk</titulo>
      <autor>Elon Musk</autor>
      <editorial>EM</editorial>
      <categoria>Biografía</categoria>
      <descripcion>El mundo de Elon</descripcion>
      <statusPrestamo>Disponible</statusPrestamo>
    </GuardarLibroRequest>
  </Body>
</Envelope>

```

Mostrar Libros

Una vez guardado el libro regresamos al wsdl para seleccionar la opción de MostrarLibros, una vez hecho eso le damos en Go para que nos devuelva todos los libros registrados.

```

POST https://microservicio-libro.herokuapp.com:443/ws Go
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <MostrarLibrosRequest xmlns="https://t4is.uv.mx/libro">[any]</MostrarLibrosRequest>
  </Body>
</Envelope>

```

La respuesta que no da es la siguiente:

```

POST https://microservicio-libro.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:MostrarLibrosResponse xmlns:ns2="https://t4is.uv.mx/libro">
      <ns2:libros>
        <ns2:id>1</ns2:id>
        <ns2:titulo>Ejemplo</ns2:titulo>
        <ns2:autor>[string]</ns2:autor>
        <ns2:editorial>[string]</ns2:editorial>
        <ns2:categoria>[string]</ns2:categoria>
        <ns2:descripcion>[string]</ns2:descripcion>
        <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>
      </ns2:libros>
      <ns2:libros>
        <ns2:id>3</ns2:id>
        <ns2:titulo>Lord Of Chaos</ns2:titulo>
        <ns2:autor>jair</ns2:autor>
        <ns2:editorial>l</ns2:editorial>
        <ns2:categoria>s</ns2:categoria>
        <ns2:descripcion>s</ns2:descripcion>
      </ns2:libros>
      <ns2:libros>
        <ns2:id>4</ns2:id>
        <ns2:titulo>Elon Musk</ns2:titulo>
        <ns2:autor>Elon Musk</ns2:autor>
        <ns2:editorial>EM</ns2:editorial>
        <ns2:categoria>Biografia</ns2:categoria>
        <ns2:descripcion>El mundo de Elon</ns2:descripcion>
        <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>
      </ns2:libros>
    </ns2:MostrarLibrosResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Modificar Libro

La siguiente acción que se hará es el de Modificar Libro, de igual manera, nos regresamos al wsdl y seleccionamos ModificarLibro, después colocamos los datos que se desea modificar de un libro en especial y le damos en Go.

```
POST https://microservicio-libro.herokuapp.com:443/ws
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <ModificarLibroRequest xmlns="https://t4is.uv.mx/libro">
      <id>1</id>
      <titulo>El código da vinci</titulo>
      <autor>Da vinci</autor>
      <editorial>DV</editorial>
      <categoria>Historia</categoria>
      <descripcion>Historia</descripcion>
      <statusPrestamo>Disponible</statusPrestamo>
    </ModificarLibroRequest>
  </Body>
</Envelope>
```

Rápidamente nos manda el mensaje de: libro modificado.

```
POST https://microservicio-libro.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:ModificarLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">
      <ns2:respuesta>Libro modificado</ns2:respuesta>
    </ns2:ModificarLibroResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Para ver si los datos se modificaron nos vamos de nuevo al wsdl y le damos en MostrarLibros y le damos en Go.

```
POST https://microservicio-libro.herokuapp.com:443/ws Go
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <MostrarLibrosRequest xmlns="https://t4is.uv.mx/libro">[any]</MostrarLibrosRequest>
  </Body>
</Envelope>
```

Finalmente, se puede observar que el libro del id 1 se modificó correctamente.

```

POST https://microservicio-libro.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:MostrarLibrosResponse xmlns:ns2="https://t4is.uv.mx/libro">
      <ns2:libros>
        <ns2:id>1</ns2:id>
        <ns2:titulo>El código da vinci</ns2:titulo>
        <ns2:autor>Da vinci</ns2:autor>
        <ns2:editorial>DV</ns2:editorial>
        <ns2:categoria>Historia</ns2:categoria>
        <ns2:descripcion>Historia</ns2:descripcion>
        <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>
      </ns2:libros>
      <ns2:libros>
        <ns2:id>3</ns2:id>
        <ns2:titulo>Lord Of Chaos</ns2:titulo>
        <ns2:autor>jair</ns2:autor>
        <ns2:editorial>l</ns2:editorial>
        <ns2:categoria>s</ns2:categoria>
        <ns2:descripcion>s</ns2:descripcion>
      </ns2:libros>
      <ns2:libros>
        <ns2:id>4</ns2:id>
        <ns2:titulo>Elon Musk</ns2:titulo>
        <ns2:autor>Elon Musk</ns2:autor>
        <ns2:editorial>EM</ns2:editorial>
        <ns2:categoria>Biografia</ns2:categoria>
        <ns2:descripcion>El mundo de Elon</ns2:descripcion>
        <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>
      </ns2:libros>
    </ns2:MostrarLibrosResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Buscar Libro

Para buscar un libro en especial, nos vamos al wsdl y seleccionamos el de BuscarLibro y nos abre la siguiente pestaña, donde, tenemos que colocar el id y título del libro a buscar, después le damos en Go.

```

POST https://microservicio-libro.herokuapp.com:443/ws Go
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <BuscarLibroRequest xmlns="https://t4is.uv.mx/libro">
      <id>1</id>
      <titulo>El código da vinci</titulo>
    </BuscarLibroRequest>
  </Body>
</Envelope>

```

Finalmente nos manda la respuesta de nuestra petición con los datos del libro que se colocó en la búsqueda.

```

POST https://microservicio-libro.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:BuscarLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">
      <ns2:libros>
        <ns2:id>1</ns2:id>
        <ns2:titulo>El código da vinci</ns2:titulo>
        <ns2:autor>Da vinci</ns2:autor>
        <ns2:editorial>DV</ns2:editorial>
        <ns2:categoria>Historia</ns2:categoria>
        <ns2:descripcion>Historia</ns2:descripcion>
        <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>
      </ns2:libros>
    </ns2:BuscarLibroResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Total de Libros

Para saber el total de libros que tenemos registrados, solo es cuestión de ir al wsdl y seleccionar la opción de TotalLibro donde nos mandará a una nueva pestaña y nuevamente le damos Go.

```
POST https://microservicio-libro.herokuapp.com:443/ws Go
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <TotalLibroRequest xmlns="https://t4is.uv.mx/libro">[any]</TotalLibroRequest>
  </Body>
</Envelope>
```

Y nos mandará el resultado de los libros registrados, en este caso solo tenemos 3 libros guardados.

```
POST https://microservicio-libro.herokuapp.com:443/ws
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:TotalLibroResponse xmlns:ns2="https://t4is.uv.mx/libro">
      <ns2:respuesta>Tu total de libros es: 3</ns2:respuesta>
    </ns2:TotalLibroResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Eliminar Libro

Otra opción que podemos realizar es el de eliminar los libros, para ello, seleccionamos la opción de EliminarLibro, donde indicamos con el id el libro que queremos eliminar, después, le damos en Go.

```
POST https://microservicio-libro.herokuapp.com:443/ws Go
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <EliminarLibroRequest xmlns="https://t4is.uv.mx/libro">
      <id>4</id>
    </EliminarLibroRequest>
  </Body>
</Envelope>
```

Para comprobar que el libro se eliminó correctamente solo nos vamos de nuevo a MostrarLibro, le damos Go y nos manda directamente la respuesta, mostrándonos los libros que se encuentran en la base de datos. Como podemos observar ya no se encuentra el libro que se acaba de eliminar.

```
POST https://microservicio-libro.herokuapp.com:443/ws Go
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:MostrarLibrosResponse xmlns:ns2="https://t4is.uv.mx/libro">
      <ns2:libros>
        <ns2:id>1</ns2:id>
        <ns2:titulo>El código da vinci</ns2:titulo>
        <ns2:autor>Da vinci</ns2:autor>
        <ns2:editorial>DV</ns2:editorial>
        <ns2:categoría>Historia</ns2:categoría>
        <ns2:descripcion>Historia</ns2:descripcion>
        <ns2:statusPrestamo>Disponible</ns2:statusPrestamo>
      </ns2:libros>
      <ns2:libros>
        <ns2:id>3</ns2:id>
        <ns2:titulo>Lord Of Chaos</ns2:titulo>
        <ns2:autor>jair</ns2:autor>
        <ns2:editorial>l</ns2:editorial>
        <ns2:categoría>s</ns2:categoría>
        <ns2:descripcion>s</ns2:descripcion>
      </ns2:libros>
    </ns2:MostrarLibrosResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Servicio público

Para poder ingresar al servicio del Público se necesitará ingresar a la siguiente liga:

<https://microservicio-publico.herokuapp.com>

Y si esta bien desplegado lo redireccionará a la siguiente página

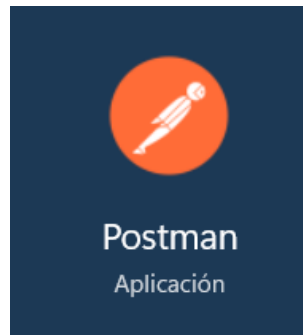
Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Fri May 27 04:23:57 GMT 2022

There was an unexpected error (type=Not Found, status=404).

Gracias a la ayuda del software Postman nos ayudará a realizar las pruebas apropiadas para verificar el buen funcionamiento de este microservicio que tiene una arquitectura de RES



Algunos de los métodos que se pueden utilizar en este microservicio son los siguientes:

GET
POST
PUT
PATCH
DELETE

Obtener usuarios

Para poder obtener los usuario que se tiene registrados en la base de datos, solo se requiere que coloquemos “usuario” y que la consulta sea por el método GET.

GET



<https://microservicio-publico.herokuapp.com/usuario>

La respuesta que nos da es la siguiente:

```
[
  {
    "id": 2,
    "nombre": "j",
    "email": "j@.com",
    "prioridad": 2,
    "domicilio": "xalapa",
    "telefono": 22
  }
]
```

Guardar usuario

Para poder guardar un usuario, solo colocamos “usuario” en la url, después, lo hacemos por el método POST y finalmente rellenamos los campos que se solicita del usuario.

POST



<https://microservicio-publico.herokuapp.com/usuario>


```
{
  "nombre": "m",
  "email": "m@.com",
  "prioridad": 2,
  "domicilio": "x",
  "telefono": 2
}
```

La respuesta que nos da es el objeto con todos sus valores:

```
{
  "id": 3,
  "nombre": "m",
  "email": "m@.com",
  "prioridad": 2,
  "domicilio": "x",
  "telefono": 2
}
```

Modificar Usuario

Para poder modificar a un usuario, se debe colocar "usuario" en la url, después, por el método POST y para poder realizar la modificación debemos de enviar los datos en formato JSON.


POST  <https://microservicio-publico.herokuapp.com/usuario>

Una vez realizado el envío de los datos, se tendrá la respuesta.

```
{
  "id": 2,
  "nombre": "p",
  "email": "p@.com",
  "prioridad": 1,
  "domicilio": "p",
  "telefono": 1
}
```

Buscar Usuario por su id

Para realizar la búsqueda de un usuario por ID será necesario completar la URL con “usuario” y el id del usuario a buscar, en este caso utilizaremos el 2 y su envío es el siguiente “/usuario/2” y todos los datos serán recibidos por el método GET.


GET  <https://microservicio-publico.herokuapp.com/usuario/2>

Si existe el usuario registrado mostrará su información de la siguiente manera

```
[
  {
    "id": 3,
    "nombre": "m",
    "email": "m@.com",
    "prioridad": 2,
    "domicilio": "x",
    "telefono": 2
  }
]
```

Eliminar Usuario por su id

Si se desea eliminar un cliente de la base de datos será necesario saber su ID para poder ser enviado a través del URL de la siguiente manera “usuario/2” utilizando el método DELETE.

DELETE  <https://microservicio-publico.herokuapp.com/usuario/2>

Si existe el usuario con ese ID será borrado de la base de datos y se recuperara un mensaje diciendo que se eliminó en usuario con el ID solicitado

Se eliminó el usuario con id 2

Servicio de préstamo

Para poder ingresar al servicio de préstamo tienes que ir a la siguiente liga:

<https://microservicio-prestamo.herokuapp.com/ws/prestamo.wsdl>

Una vez hecho esto te abrirá lo siguiente:

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:sch="https://t4is.uv.mx/prestamo" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="https://t4is.uv.mx/prestamo" targetNamespace="https://t4is.uv.mx/prestamo">
  <wsdl:types>
    <xsi:schema xmlns:xsi="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" targetNamespace="https://t4is.uv.mx/prestamo">
      <!-- Realizar Prestamo -->
      <xsi:element name="GuardarPrestamoRequest">
        <xsi:complexType>
          <xsi:sequence>
            <xsi:element name="finicio" type="xs:string"/>
            <xsi:element name="ffin" type="xs:string"/>
            <xsi:element name="nomcliente" type="xs:string"/>
            <xsi:element name="titulolibro" type="xs:string"/>
            <xsi:element name="status" type="xs:string"/>
            <xsi:element name="responsable" type="xs:string"/>
          </xsi:sequence>
        </xsi:complexType>
      </xsi:element>
      <xsi:element name="GuardarPrestamoResponse">
        <xsi:complexType>
          <xsi:sequence>
            <xsi:element name="respuesta" type="xs:string"/>
          </xsi:sequence>
        </xsi:complexType>
      </xsi:element>
      <!-- Renovar Prestamo -->
      <xsi:element name="RenovarPrestamoRequest">
        <xsi:complexType>
          <xsi:sequence>
            <xsi:element name="id" type="xs:int"/>
            <xsi:element name="finicio" type="xs:string"/>
            <xsi:element name="ffin" type="xs:string"/>
            <xsi:element name="nomcliente" type="xs:string"/>
            <xsi:element name="titulolibro" type="xs:string"/>
            <xsi:element name="status" type="xs:string"/>
            <xsi:element name="responsable" type="xs:string"/>
          </xsi:sequence>
        </xsi:complexType>
      </xsi:element>
    </xsi:schema>
  </wsdl:types>

```

Para hacer uso de este servicio tienes que tener la herramienta de Wizdler que analiza los archivos WSDL y genera mensajes SOAP



Una vez hecho esto, lo siguiente es seleccionar el de GuardarPrestamo y ingresamos los datos solicitados

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <GuardarPrestamoRequest xmlns="https://t4is.uv.mx/prestamo">
      <finicio>3 de junio de 2022</finicio>
      <ffin>5 de junio de 2022</ffin>
      <nomcliente>Luis</nomcliente>
      <titulolibro>En las nubes</titulolibro>
      <status>Prestado</status>
      <responsable>Arisbeth</responsable>
    </GuardarPrestamoRequest>
  </Body>
</Envelope>
```

Le damos en Go y nos mandara el siguiente mensaje:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:GuardarPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:respuesta>Hola Luis acaba de apartar el libro En las nubes hasta 5 de junio de 2022</ns2:respuesta>
    </ns2:GuardarPrestamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Después, verificamos que los datos se encuentren en la base de datos, para ello, nos vamos al WSDL para seleccionar el de MostrarPrestamos

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

Le damos en Go y nos muestra el préstamo que fue realizado.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:MostrarPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:prestamo>
        <ns2:id>1</ns2:id>
        <ns2:finicio>1 de junio 2022</ns2:finicio>
        <ns2:ffin>10 de junio 2022</ns2:ffin>
        <ns2:nomcliente>p</ns2:nomcliente>
        <ns2:titulolibro>libro1</ns2:titulolibro>
        <ns2:status>disponible</ns2:status>
        <ns2:responsable>m</ns2:responsable>
      </ns2:prestamo>
      <ns2:prestamo>
        <ns2:id>2</ns2:id>
        <ns2:finicio>3 de junio de 2022</ns2:finicio>
        <ns2:ffin>5 de junio de 2022</ns2:ffin>
        <ns2:nomcliente>Luis</ns2:nomcliente>
        <ns2:titulolibro>En las nubes</ns2:titulolibro>
        <ns2:status>Prestado</ns2:status>
        <ns2:responsable>Arisbeth</ns2:responsable>
      </ns2:prestamo>
    </ns2:MostrarPrestamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Lo siguiente que realizaremos es buscar un préstamo por cliente, nos regresamos nuevamente al WSDL y seleccionamos PrestamoPorCliente, introducimos el nombre del cliente y le damos en Go

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <PrestamoPorClienteRequest xmlns="https://t4is.uv.mx/prestamo">
      <nomCliente>Luis</nomCliente>
    </PrestamoPorClienteRequest>
  </Body>
</Envelope>
```

Nos muestra como respuesta los datos del préstamo del cliente que introducimos

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:PrestamoPorClienteResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:prestamo>
        <ns2:id>2</ns2:id>
        <ns2:finicio>3 de junio de 2022</ns2:finicio>
        <ns2:ffin>5 de junio de 2022</ns2:ffin>
        <ns2:nomcliente>Luis</ns2:nomcliente>
        <ns2:titulolibro>En las nubes</ns2:titulolibro>
        <ns2:status>Prestado</ns2:status>
        <ns2:responsable>Arisbeth</ns2:responsable>
      </ns2:prestamo>
    </ns2:PrestamoPorClienteResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Vamos a renovar un préstamo, para hacer esto nos vamos de nuevo al WSDL donde seleccionaremos el de RenovarPréstamo, para ello, introducimos el id y los datos a renovar, le damos en Go

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <RenovarPréstamoRequest xmlns="https://t4is.uv.mx/prestamo">
      <id>2</id>
      <finicio>3 de junio de 2022</finicio>
      <ffin>15 de junio de 2022</ffin>
      <nomcliente>Luis</nomcliente>
      <titulolibro>En las nubes</titulolibro>
      <status>Prestado</status>
      <responsable>Arisbeth</responsable>
    </RenovarPréstamoRequest>
  </Body>
</Envelope>
```

Una vez actualizado la fecha de entrega nos muestra el siguiente mensaje:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:RenovarPréstamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:respuesta>Ha renovado el préstamo del libro En las nubes hasta el 15 de junio de 2022</ns2:respuesta>
    </ns2:RenovarPréstamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Por último, veremos la opción de ver el estado de un libro, para realizar esta acción nos vamos de nuevo al WSDL y seleccionamos el de StatusPréstamo, introducimos el id y el título del libro, le damos en Go.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <StatusPréstamoRequest xmlns="https://t4is.uv.mx/prestamo">
      <id>2</id>
      <titulolibro>En las nubes</titulolibro>
    </StatusPréstamoRequest>
  </Body>
</Envelope>
```

Nos manda como respuesta que el libro “En las nubes” está prestado.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:StatusPrestamoResponse xmlns:ns2="https://t4is.uv.mx/prestamo">
      <ns2:respuesta>El status del libro: En las nubes es: Prestado</ns2:respuesta>
    </ns2:StatusPrestamoResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Dockerfile del microservicio-libro

```
From rrojano/jdk8
workdir /app
#expose 8080
cmd ["/app/ejecutar.sh"]
add app/Libro-0.0.1-SNAPSHOT.jar /app/Libro-0.0.1-SNAPSHOT.jar
add ejecutar.sh /app/ejecutar.sh
run chmod 755 /app/ejecutar.sh
```

Script ejecutar.sh del microservicio-libro

```
#!/bin/sh
/usr/bin/java -jar -Dserver.port=$PORT Libro-0.0.1-SNAPSHOT.jar
```

Dockerfile del microservicio-publico

```
From jairvr/jdk8
workdir /app
#expose 8080
cmd ["/app/ejecutar.sh"]
add app/demo-0.0.1-SNAPSHOT.jar /app/demo-0.0.1-SNAPSHOT.jar
add ejecutar.sh /app/ejecutar.sh
run chmod 755 /app/ejecutar.sh
```

Script ejecutar.sh del microservicio-publico

```
#!/bin/sh
/usr/bin/java -jar -Dserver.port=$PORT demo-0.0.1-SNAPSHOT.jar
```

Dockerfile del microservicio-prestamo

```
From rrojano/jdk8
workdir /app
#expose 8080
cmd ["/app/ejecutar.sh"]
add app/Prestamo-0.0.1-SNAPSHOT.jar /app/Prestamo-0.0.1-SNAPSHOT.jar
add ejecutar.sh /app/ejecutar.sh
run chmod 755 /app/ejecutar.sh
```

Script ejecutar.sh del microservicio-prestamo

```
#!/bin/sh
/usr/bin/java -jar -Dserver.port=$PORT Prestamo-0.0.1-SNAPSHOT.jar
```

Proyecto publicado en github

link del repositorio: <https://github.com/JairVqz/ProyectoT4IS-Equipo5>

Proyecto desplegado

Microservicio Libro

<https://microservicio-libro.herokuapp.com/ws/libro.wsdl>

Microservicio Publico

<https://microservicio-publico.herokuapp.com/usuario>

Microservicio Prestamo

<https://microservicio-prestamo.herokuapp.com/ws/prestamo.wsdl>