Una pantalla de una computadora

Descripción generada automáticamente

Contenido

[Introducción 3](#_Toc104513067)

[Motivación 3](#_Toc104513068)

[Problemática 3](#_Toc104513069)

[Solución 5](#_Toc104513070)

[ Servicio de Tickets o Boletos 5](#_Toc104513071)

[ Servicio de dulcería: 5](#_Toc104513072)

[ Servicio de Empleados: 5](#_Toc104513073)

[Costos 5](#_Toc104513074)

[ Creación de repositorio compartido 5](#_Toc104513075)

[ Servicios de nube 5](#_Toc104513076)

[ Hosting 6](#_Toc104513077)

[ Peticiones para testear APIs de tipo REST 6](#_Toc104513078)

[ Mano de obra 6](#_Toc104513079)

[Diagrama de despliegue 7](#_Toc104513080)

[Documentación de API SOAP y REST. 8](#_Toc104513081)

[Microservicio de Tickets y Boletos 8](#_Toc104513082)

[EndPoint 8](#_Toc104513083)

[Microservicio de Empleados 10](#_Toc104513084)

[EndPoint 10](#_Toc104513085)

[Parámetros de recepción 14](#_Toc104513086)

[Microservicio Empleados 14](#_Toc104513087)

[Parámetros devueltos 18](#_Toc104513088)

[Microservicio Empleados 18](#_Toc104513089)

[Plan de pruebas 22](#_Toc104513090)

[Proyecto publicado en github 22](#_Toc104513091)

[Microservicio Tickets y Boletos 22](#_Toc104513092)

[Microservicio Empleados 22](#_Toc104513093)

[Proyecto publicado en Heroku 22](#_Toc104513094)

[Microservicio Tickets y Boletos 22](#_Toc104513095)

[Microservicio Empleados 22](#_Toc104513096)

# Introducción

En el presente proyecto de Tecnologías para la integración de soluciones, tiene como objetivo evaluar y adaptar los conocimientos adquiridos durante el curso, para ello se tiene planeado realizar un servicio de API´S con distintos Microservicios, todo enfocado a un Cine para resolver algunos servicios que requiera, en el cual emplearemos las tecnologías que son SOAP y REST para su desarrollo.

Nuestro proyecto será realizado siguiendo los puntos solicitados a evaluar por parte del profesor Rojano, en este caso no nos enfocaremos como tal en el fronted, solo sera por la parte del backend. En este servicio de cine se pretende poder hacer consumo de los servicios que puede ofrecer un cine, tenemos pensado que el consumo del servicio sea para los cliente y también para que el cine pueda administrar algunos detalles.

# Motivación

La motivación de nuestro proyecto surge con la idea de ayudar a la facilitación y buen manejo administrativo de un cine cualquiera con ciertas funciones como compra de boletos o tickets, poder comprar facilmente dulces y administrar a los empleados del cine.

# Problemática

Una de las problematicas que se encontraron fue en la parte administrativa de un cine por ende proponemos que nuesro servicio sirva para tras aspectos que ayuden a solucionar la organización, por ejemplo en la parte de la dulceria en donde le permitira al usuario comprar dulces que esten disponibles en el inventario y solo llevar un ticket al Cine de esta manera poder evitar largas filas y ahorrar tiempo.

Por otra parte administrativamente el poder comprar boletos y ver la cartelera disponible en el sistema ahorrara bastante tiempo y permitira que este controlado el flujo de personas en el Cine.Por ultimo el manejamiento de empleados para una adecuada organización interna del personal.

# Solución

Para la solución de este problema se tiene planeado realizar un servicio de API el cual esta dividido en microservicios siendo estos:

## **Servicio de Tickets o Boletos:** Este microservicio permitirá que el que se revise la cartelera disponible, ver las funciones que se haran por cada pelicula, para ver la cartelera se tendran atributos como (IdFuncion, Nombre de pelicula, Hora, Fecha, Precio, Clasificación y Asientos disponibles), tambien se podra comprar boletos y cambiar u modificar la función.

* **Servicio de dulcería:** Este microservicio se incorporar la opcion de comprar dulces, ver el menu, el inventario y modificar el mismo
* **Servicio de Empleados:** En ese microservicio se empleará un CRUD con las siguientes opciones: Dar de Alta Empleados, Ver Empleados, Eliminar Empleados y Modificar los Empleados. Cada apartado tendrá sus distintos atributos

# Costos

Las tecnologias utilizadas para nuestro proyecto que nos permitieron deducir costos son las siguientes:

## **Creación de repositorio compartido:**

Utlizamos la tecnología de GitHub para la creación de nuestros repositorio y poder todos apoyar y codificar subiendo nuestros combios. Esta tecnología es gratuita por ende este recurso no nos costo.

## **Servicios de nube**

En este aspecto para el despliegue de nuestros microservicios utilizaremos Heroku en su versión gratuita, que nos da el almacenamiento para soportar nuestro servicio, lo cual no generó gastos adicionales.

## **Hosting:**

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

## **Peticiones para testear APIs de tipo REST**

Para las peticiones la tecnología que ocupamos en post man es de uso gratuito y nos va a servir para consumir un recurso de un servicio web desde un cliente de manera grafica.

## **Mano de obra**

Para este punto no fue necesario el pago de sueldos ni la contratación debido a que somos cuatro desarolladores trabajando en especificamente este servicio el cual fue dividido por actividades y por un lapzo de tiempo considerable no se requiere mano de obra extra.

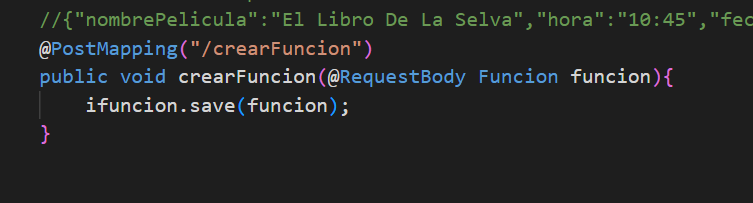
# Diagrama de despliegue

# Documentación de API SOAP y REST.

## **Microservicio de Tickets y Boletos**

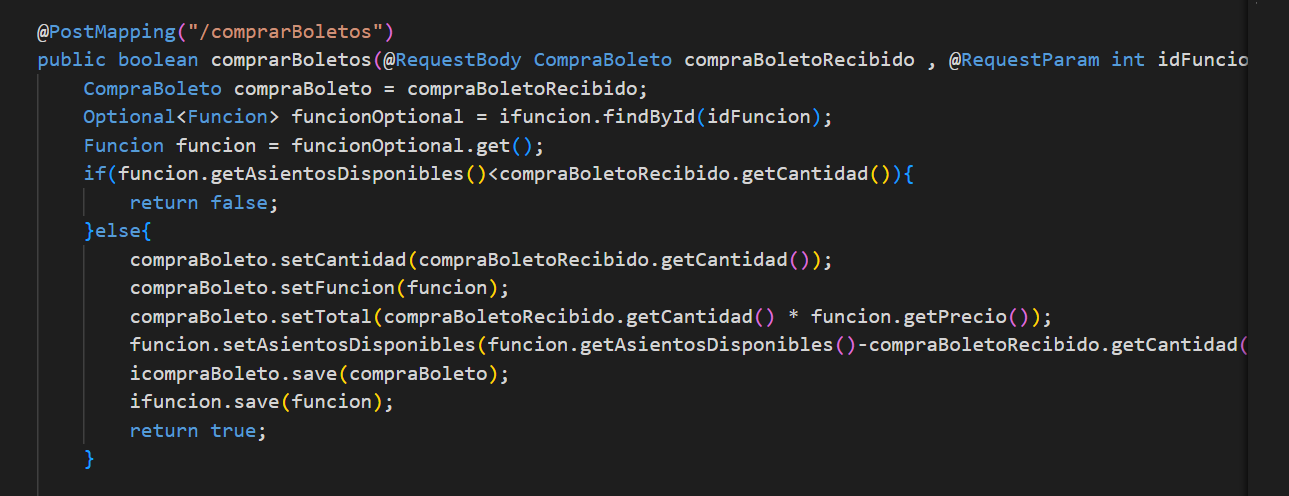
### **EndPoint**

**“/crearfunción”**

<https://microservicio-funcion.herokuapp.com/crearFuncion>

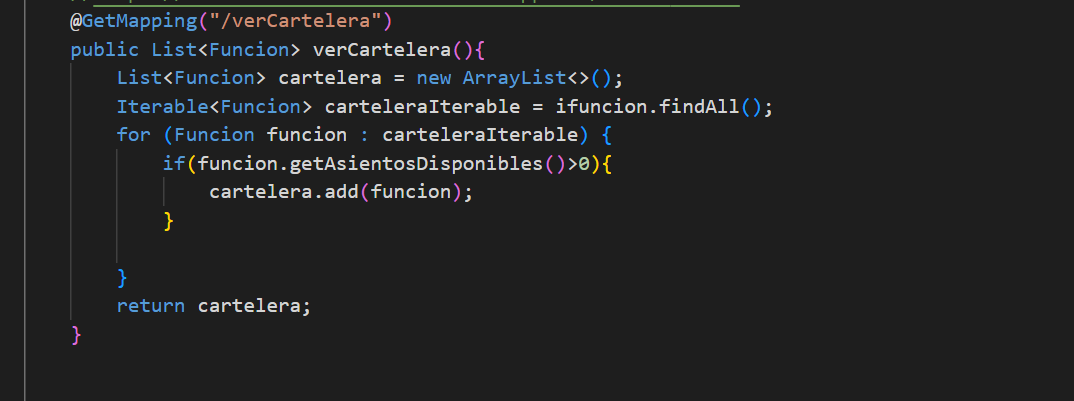
**“/comprarBoletos”**

Para el recurso de comprar un boleto para alguna funcion se hace uso de un envio post pero también se necesita pasar un parametro por la URI aquí se pasa el id de la funcion de la cual se quiere comprar boletos)

<https://microservicio-funcion.herokuapp.com/comprarBoletos?idFuncion=1>

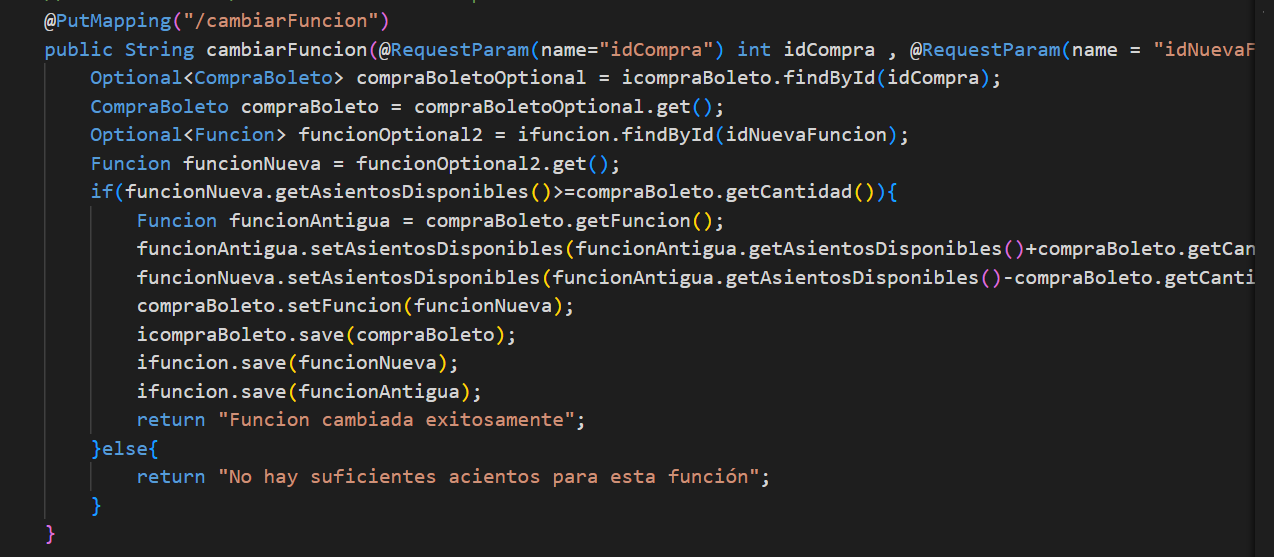
**“/verCartelera”**

Para este recurso solamente en necesario consumirlo con la siguiente URL:

    <https://microservicio-funcion.herokuapp.com/verCartelera>

**“/cambiarFunción”**

Aquí no es necesario hacer uso de un archivo JSON ya que lo que requerimos son parametros, el id de la compra de los boletos y la nueva funcion a la que se desea cambair.

 localhost:8080/cambiarFuncion?idCompra=2&idNuevaFuncion=7

## **Microservicio de Empleados**

### **EndPoint**

**“Dar de Alta Empleado”**

Para poder de alta un empleado, se hace uso de lo que es el request y response en relación a un archivo XSD llamado Empleados. Lo que hacemos es entrar a la siguiente pagina en heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl>

y despues hacemos uso de la extensión wizdler en donde daremos clic a la opción “RegistrarEmpleado” y llenaremos los campos correspondientes que son: Nombre, Apellido, Fecha de nacimiento, Direccion, Telefono, Sexo y Estado civil; todos estos datos los recibiremos con request y los guardaremos y mostraremos con response.

Todos esos datos los guardaremos en una base de datos en la nube la cual es “Clever cloud”

Texto

Descripción generada automáticamente

**“Ver Empleados”**

En este apartado lo que se podrá hacer es la visualización de los distintos empleados registrados en la base de datos Clever Cloud, en este caso no ingresaremos ningún dato por parte de request, sino nomas haremos la petición para extraer los datos de la base de datos y mostrarlos en pantalla con ayuda response y del wizdler

<https://cine-empleados.herokuapp.com/ws/Empleados.wsdl>

Captura de pantalla de computadora

Descripción generada automáticamente

**“Modificar Empleado”**

Para poder modificar un empleado, se hace uso de lo que es el request y response en relación a un archivo XSD llamada Empleados. Lo que hacemos es entrar a la siguiente pagina en heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl>

Y despues hacemos uso de la extensión wizdler en donde daremos clic a la opción “ModificarEmpleado” y llenaremos los campos correspondientes, introduciendo el id del empleado que le deseamos modificarle los datos y los datos nuevos a reemplazar como son: Nombre, Apellido, Fecha de nacimiento, Direccion, Telefono, Sexo y Estado civil; todos estos datos los recibiremos con request y de hay se guardan el base de datos de la nube Clever cloud y con response mostramos los datos ya actualizados.

Texto

Descripción generada automáticamente

**“Eliminar Empleado”**

Para poder Eliminar un empleado, se hace uso de lo que es el request y response en relación a un archivo XSD llamado Empleados. Lo que hacemos es entrar a la siguiente pagina en heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl>

Y despues hacemos uso de la extensión wizdler en donde daremos clic a la opción “EliminarEmpleado” y llenaremos el campo correspondiente que en este caso es el id del empleado que deseamos eliminar. Despues de haber eliminado el empleado deseado, se nos mostrara con el response la lista de empleados que se encuentra en la base de datos Clever Cloud menos el empleado eliminado.

Texto

Descripción generada automáticamente

# Parámetros de recepción

**“/crearfunción”**

**“/comprarBoletos”**

**“/verCartelera”**

**“/cambiarFunción”**

## **Microservicio Empleados**

**RegistrarEmpleado (Request)**

Para poder registrar un Empleado se requerirá llenar los siguientes campos en el código XSD con ayuda de la extensión Wizdler, los campos son: Nombre, Apellido, Fecha de nacimiento, Dirección, Teléfono, Sexo y Estado Civil. Todos como campos String.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Código Empleados.xsd

Captura de pantalla de computadora

Descripción generada automáticamente

Código RegistrarEmpleado (Request)

**Empleados (Request)**

En este apartado no se requiere como tal llenar un campo, sino solo darle al botón “Go” para poder mandar la petición de que se nos muestre la lista de empleados que están registrados en la Base de datos de Clever Cloud

Captura de pantalla de computadora

Descripción generada automáticamente

Código Empleados.xsd

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Código Empleados (request)

**ModificarEmpleado (Request)**

Para poder modificar un Empleado se requerirá llenar los siguientes campos en el código XSD con ayuda de la extensión Wizdler, los campos son: id, NuevoNombre, NuevoApellido, NuevaFechaNacimiento, NuevaDireccion, NuevoTelefono, NuevoSexo y NuevoEstadoCivil. Todos estos campos serán String menos el id, el será un Int (Entero)

En el caso del campo del id, hay debemos ingresar el id del Empleado al que vamos a modificar sus datos

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Código Empleados.xsd

Captura de pantalla de computadora

Descripción generada automáticamente

Código ModificarEmpleado (resquest)

**EliminarEmpleado (Request)**

Para poder Eliminar un Empleado se requerirá llenar el siguiente campo en el código XSD con ayuda de la extensión Wizdler, el campo es el id del Empleado que deseamos eliminar.

Captura de pantalla de computadora

Descripción generada automáticamente

Código Empleados.xsd

Captura de pantalla de computadora

Descripción generada automáticamente

Código EliminarEmpleado (request)

# Parámetros devueltos

## **Microservicio Empleados**

**RegistrarEmpleado (Response)**

La respuesta que recibimos después de aplicar RegistrarEmpleado (request), es un mensaje mostrando todos los campos con los datos del nuevo empleado que registramos en la base de datos y el estatus de la operación.

Captura de pantalla de computadora

Descripción generada automáticamente

Código Empleados.xsd

Captura de pantalla de computadora

Descripción generada automáticamente

Código RegistrarEmpleado (Response)

**Empleados (response)**

La respuesta que recibimos en este apartado es la lista de empleados que se encuentra almacenada en la base de datos clever cloud y el estatus de la operación.

Captura de pantalla de computadora

Descripción generada automáticamente

Código Empleados.xsd

Captura de pantalla de computadora

Descripción generada automáticamente

Código Empleados (response)

**ModificarEmpleado (Response)**

La respuesta que recibimos después de aplicar ModificarEmpleado (request), es un mensaje mostrando todos los campos de un empleado con nuevos datos que se modificaron en la base de datos Clever cloud. También recibimos el estatus de la operación.

Captura de pantalla de computadora

Descripción generada automáticamente

Código Empleados.xsd

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Código ModificarEmpleado (response)

**EliminarEmpleado (Response)**

La respuesta que recibimos es la lista de empleados ya actualizada desde la base de datos Clever Cloud, mostrando los datos de cada empleado mediante los distintos campos como son el nombre, apellido, etc.

También lo recibimos como respuesta es el estatus de la operación.

Texto

Descripción generada automáticamente

Código Empleados.xsd

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteCaptura de pantalla de computadora

Descripción generada automáticamente

Código EliminarEmpleado (response)

# Plan de pruebas

El plan de pruebas aquí presentado se realizó con el propósito de identificar posibles fallas y errores de nuestro servicio web “Cine”, también uno de los propósitos es verificar el correcto funcionamiento de cada uno de los microservicios y las funcionalidades que lo componen a cada uno de ellos. De manera detallada podemos resaltar que la prueba se realizo a todos los microservicios incluidos en nuestro servicio web “Cine”, por cada funcionalidad de cada microservicio se realizo una prueba y en su gran mayoría los resultados obtenidos fueron exitosos.

## **Microservicio Empleados**

### **Recursos**

#### **Requerimientos de entornos – Hardware**

Hardware: Una computadora con acceso a un navegador

Red: Conexión a internet wifi

#### **Requerimientos de entornos – Software**

Software: extensión de wizdler en nuestro navegador (para el caso de SOAP)

### **Planificación y Organización**

#### **Procedimientos para las pruebas**

La prueba se realizó siguiendo una serie de pasos dependiendo de la funcionalidad del microservicio.

|  |  |
| --- | --- |
| Fecha | 26 de mayo del 2022 |
| Titulo | Registrar Empleado |
| Descripción | El usuario registra un empleado llenando los campos correspondientes |
| Condiciones de Entrada | * El usuario debe entrar a link de heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl> * Debe usar la extensión wizdler y seleccionar la funcionalidad “RegistrarEmpleado” * Llenar los campos correspondientes en la parte de request * Y darle clic en el botón “GO” |
| Resultado Esperado | El servicio debe guardar los datos introducidos en la base de datos Clever Cloud y mostrarlos en pantalla en la parte de response. |
| Resultado Obtenido | El servicio guarda los datos en la base de datos Clever cloud, muestra esos datos en la pantalla en la parte de response y muestra un estatus de la operación. |

|  |  |
| --- | --- |
| Fecha | 26 de mayo del 2022 |
| Titulo | Empleados |
| Descripción | El usuario podrá ver la lista de empleados junto con sus datos, que se encuentran guardados en la BD |
| Condiciones de Entrada | * El usuario debe entrar a link de heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl> * Debe usar la extensión wizdler y seleccionar la funcionalidad “Empleados” * No se debe llenar ningún campo, solo darle clic al botón “GO” que esta en la pantalla request |
| Resultado Esperado | El servicio nos debe mostrar la lista de empleados con sus respectivos datos en la pantalla response |
| Resultado Obtenido | El servicio nos muestra la lista de empleados con sus respectivos datos y el estatus de la operación, todo en la pantalla response. |

|  |  |
| --- | --- |
| Fecha | 26 de mayo del 2022 |
| Titulo | Modificar Empleado |
| Descripción | EL usuario podrá modificar los datos de cualquier empleado que se encuentre registrado en la base de datos Clever Cloud |
| Condiciones de Entrada | * El usuario debe entrar a link de heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl> * Debe usar la extensión wizdler y seleccionar la funcionalidad “ModificarEmpleado” * Se deben llenar todos los campos con los datos nuevos que se desean reemplazar, sin dejar atrás que lo mas importante es ingresar el id del empleado al que le vamos a modificar sus datos. Todo estos se hace en la pantalla de request * Y por ultimo le damos clic al boton “GO” |
| Resultado Esperado | El servicio nos debe mostrar los campos del empleado que modificamos ya con los datos nuevos y en la BD actualizados. Todo esto en la pantalla response. |
| Resultado Obtenido | El servicio nos muestra los campos del empleado ya modificados, con lo datos nuevos y en la base de datos actualizados. Todo esto se nuestra en la pantalla response. |

|  |  |
| --- | --- |
| Fecha | 26 de mayo del 2022 |
| Titulo | Eliminar Empleado |
| Descripción | El usuario podrá eliminar cualquier empleado que se encuentre en la BD por medio de su id |
| Condiciones de Entrada | * El usuario debe entrar a link de heroku: <https://cine-empleados.herokuapp.com/ws/Empleados.wsdl> * Debe usar la extensión wizdler y seleccionar la funcionalidad “EliminarEmpleado” * Se debe introducir el id del empleado que deseamos eliminar, todo esto en la pantalla request * Y damos clic al boton “GO” |
| Resultado Esperado | El servicio nos debe mostrar la lista de empleados guardada en la base de datos menos el empleado que eliminamos |
| Resultado Obtenido | El servicio nos muestra la lista de empleados guardada en la base de datos Clever Cloud menos el empleado que eliminamos. Todo esto se nos muestra en la pantalla response. |

## **Dockerfile del Microservicio Empleados**

**Código del archivo Dockerfile**

From rrojano/jdk8

workdir /app

#expose 8080

cmd ["/app/ejecutar.sh"]

add app/EmpleadosBd-0.0.1-SNAPSHOT.jar /app/EmpleadosBd-0.0.1-SNAPSHOT.jar

add ejecutar.sh /app/ejecutar.sh

run chmod 755 /app/ejecutar.sh

## **Ejecutar del Microservicio Empleados**

**Codigo del archivo ejecutar.sh**

#!/bin/sh

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

# Proyecto publicado en github

## **Microservicio Tickets y Boletos**

[juxo321/Cine (github.com)](https://github.com/juxo321/Cine)

## **Microservicio Empleados**

<https://github.com/bryanUCP/Cine_Empleados.git>

# Proyecto publicado en Heroku

## **Microservicio Tickets y Boletos**

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

## **Microservicio Empleados**

<https://cine-empleados.herokuapp.com/ws/Empleados.wsdl>