



## Contratos

Users	Http	Uri	Descripción
	Get	<a href="#">/users</a>	Obtiene todos los usuarios
	Post	<a href="#">/users</a>	Crea un usuario
	Delete	<a href="#">users/{id}</a>	Elimina a un usuario por el id
Movies	Http	Uri	Descripción
	Get	<a href="#">/movies</a>	Obtiene todas las películas
	Post	<a href="#">/movies</a>	Crea una película
	Get	<a href="#">/movies/{id}</a>	Obtiene una película por el id
	Delete	<a href="#">/movies/{id}</a>	Elimina una película por el id
Showtimes	Http	Uri	Descripción
	Get	<a href="#">/showtimes</a>	Obtiene todas las programaciones
	Post	<a href="#">/showtimes</a>	Crea una programación
	Get	<a href="#">/showtimes/{id}</a>	Obtiene una programación por el id
	Put	<a href="#">/showtimes/{id}</a>	Actualiza una programación por el id
Bookings	Http	Uri	Descripción
	Get	<a href="#">/bookings</a>	Obtiene todas las reservaciones
	Post	<a href="#">/bookings</a>	Crear una reservación.
	Get	<a href="#">/bookings/{id}</a>	Obtiene una reserva por el id
	Delete	<a href="#">/bookings/{id}</a>	Elimina una reserva por el id
	Get	<a href="#">/bookings/{userid}</a>	Obtiene la reserva por el userid

- No se puede eliminar una película si tiene programaciones o reservas asociadas.
- No se puede eliminar un usuario si tiene reservas asociadas

## Consideraciones

- Se deben aplicar los conceptos vistos en clase
- Para cada entidad del dominio crear un microservicio
- Se deben validar los campos de las entidades según las reglas
- Crear servidor de configuración centralizado (Config-server)
- Las configuraciones pueden estar en repositorio local o en la nube
- Crear un servidor para registro y localización de instancias de microservicios (Eureka server)
- Utilizar Feign para generar el cliente de servicio REST al menos en la reserva, para la programación es opcional.
- Hacer una prueba unitaria a la capa de repositorio y la capa de servicio (lógica) en al menos dos microservicios.
- Utilizar el patrón circuit breaker en el microservicio de reserva
- Implementar Gateway con spring cloud
- Implementar un servicio de administración con Spring boot admin
- Persistir los datos en una base de datos (H2, postgresql, mysql)
- Implementar capa DTO (0.20)
- Desplegar los microservicios en la nube Heroku/AWS/etc (0.20)
- Hacer una librería (Commons) para cosas comunes de los microservicios. (0.20)
- El taller se puede hacer en parejas
- Fecha entrega: 23-30 Julio
- Fecha socialización: 23-30 Julio