

## **Desafío - Servicio Control de Destinos**

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el .zip en el LMS.
- Desarrollo desafío:
  - o El desafío se debe desarrollar de manera Individual

#### Descripción

Crear una aplicación que:

- Permita el ingreso de pasajeros con sus respectivos datos, además de listarlos.
- Permita ingresar destinos (ciudades) y generar una lista.
- Al momento de crear un destino, se debe poder asignar a un usuario, con el objetivo de identificar destinos por usuarios.

### Requerimientos

- 1. Usar Spring Initializr para generar proyecto con las respectivas dependencias.
- Crear un modelo <u>Pasajero</u> en carpeta model, la cual debe contener anotaciones de JPA, tener una relación de 1:N con <u>Destino</u> y contar con los siguientes atributos:
  - ID
  - RUT
  - Nombre
  - Apellido
  - Edad
  - CiudadNatal
- 3. Crear un modelo *Destino* en carpeta *model*, la cual debe contener anotaciones de *JPA*, tener una relación de N:1 con *Pasajero* y contar con los siguientes atributos:
  - ID
  - Ciudad
  - Pais
  - Fecha

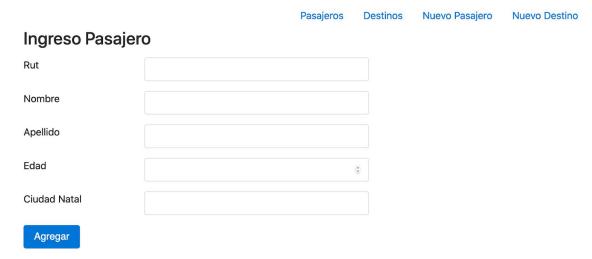


- 4. Crear una interfaz llamada DestinoRepository en la carpeta repository.
- 5. Crear una interfaz llamada DestinoService en la carpeta service.
  - Método save para crear destino.
  - Método *findAll* para listar destinos.
- 6. Crear una clase llamada *DestinoServiceImpl* en la carpeta *service*.
  - Método save para crear destino.
  - Método findAll para listar destinos.
- 7. Crear una interfaz llamada *PasajeroRepository* en la carpeta *repository*.
- 8. Crear una interfaz llamada PasajeroService en la carpeta service.
  - Método save para crear pasajero.
  - Método findAll para listar pasajeros.
- 9. Crear una clase llamada PasajeroServiceImpl en la carpeta service.
  - Método save para crear pasajeros.
  - Método findAll para listar pasajeros.
- 10. Crear una clase DestinoController.
  - Método *crearDestino* para mostrar formulario de destino. (GET)
  - Método *crearDestino* para crear destino. (POST)
  - Método destinos para mostrar vista de destinos.
- 11. Crear una vista *crear-destino.html* en carpeta *resources/templates* para agregar destino a pasajero.
- 12. Crear una vista destinos.html en carpeta resources/templates para listar destinos registrados.
- 13. Crear una clase PasajeroController.
  - Método crearPasajero para mostrar formulario de pasajero. (GET)
  - Método crearPasajero para crear pasajero. (POST)
  - Método pasajeros para mostrar vista de pasajeros.
- 14. Crear una vista *crear-pasajero.html* en la carpeta *resources/templates* para agregar pasajero.
- 15. Crear una vista *pasajeros.html* en carpeta *resources/templates* para listar pasajeros registrados.

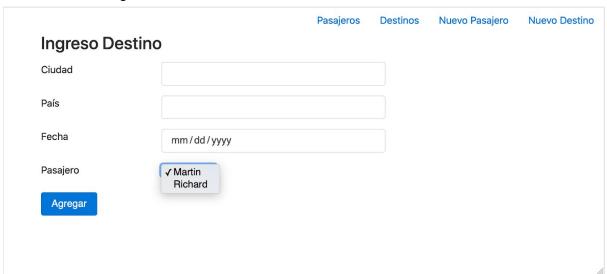


### Consideraciones

El formulario de ingreso de pasajero debe lucir así:



El formulario de ingreso de destino debe lucir así:





El sitio que muestra la lista de destinos ingresados debe lucir así:

Pasajeros	Destinos	Nuevo Pasajero	Nuevo Destino	

## **Destinos**

Ciudad	País	Fecha	Pasajero	
Laussane	Suiza	2019-11-25	Martin	

El sitio que muestra la lista de pasajeros ingresados debe lucir así:

Pasajeros	Destinos	Nuevo Pasajero	Nuevo Destino	

# Pasajeros

Rut	Nombre	Apellido	Edad	Ciudad Natal	Destinos
1-9	Martin	Odersky	61	ldN	Laussane
2-6	Richard	Hickey	56	IdNN	