

**INTERNATIONAL AIRPORT SYSTEM**

**ALEJANDRO VARELA FRANCO – A00369019 -**

**JUAN FELIPE SINISTERRA FAJARDO - A00159604 -**

**JUAN DAVID BALLESTEROS VALENCIA - A00306456 -**

**Proyecto Final del Curso**

**Profesor de Algoritmos y Programación 2**

**Juan Manuel Reyes García**

**UNIVERSIDAD ICESI**

**ALGORITMOS Y PROGRAMACIÓN 2**

**CALI, COLOMBIA**

**2021**



## ENUNCIADO

La ciudad de Madrid ha decidido construir un nuevo aeropuerto exclusivo para vuelos internacionales debido a la emergencia sanitaria por COVID-19. Dentro de los elementos del proyecto, está el desarrollo de un sistema que les permita gestionar de manera eficiente, la compra y venta de tiquetes, las aerolíneas, el personal administrativo, contemplar el flujo de personas en el área de migración y visualizar los vuelos activos en tiempo real.

Cuatro tipos de usuario utilizarán el sistema y podrán realizar diferentes acciones dentro del mismo. A continuación se describen las acciones de cada uno:

### **Administrador del aeropuerto**

El administrador del aeropuerto es el encargado de gestionar (crear, modificar y eliminar) los diferentes perfiles de usuario de los servidores de las diferentes entidades aeroportuarias (supervisor de la torre de control, agente de migración y administrador de aerolínea).

### **Supervisor de la torre de control**

El supervisor de la torre de control es el encargado de monitorear el estado actual de los vuelos, monitoreando si algún vuelo se encuentra en crisis. Además de revisar el estado de las pistas y la torre de control, habilitando el mantenimiento preventivo de las mismas, para lo cual debe notificar el tiempo de mantenimiento y programar la salida de vuelos.

Es importante que el supervisor pueda acceder y revisar todos los vuelos que están programados dentro de un rango de 24 horas y dejar asignado los vuelos a una pista con la ayuda de un algoritmo.

### **Agente de migración**

El agente de migración necesita poder observar un tablero de información que refleje el flujo de personas por el área de migración. Dicho tablero muestra la cola de personas que están esperando a pasar por el área y el tiempo que falta para que todos pasen, la cantidad de personas detenidas por ser menores de

edad y viajar sin acompañante, no tener la prueba negativa del Covid-19 o ser buscados por crímenes. También se deberá poder observar las personas que han podido pasar por migración, dando un total general y otro diferenciado por género.

El agente de migración también podrá generar reportes de los indicadores en cualquier momento, segmentado por fecha el reporte.

### **Administrador de aerolínea**

La persona que tenga un usuario de administrador de aerolínea generado por el administrador del aeropuerto es encargado primeramente de completar el perfil de la aerolínea que representa, asignar destinos, aviones y personal con el cual cuenta la aerolínea para operar. Después de tener el perfil completo el administrador podrá generar vuelos y asignar el precio, posterior a esto los clientes podrán ver la oferta de la aerolínea en la plataforma.

Al crear un nuevo vuelo es necesario asignar el destino, la duración en minutos, los pilotos y asistentes asignados con la verificación que estos no se encuentren asignados a otro vuelo en ese momento. También es necesario el precio de los tiquetes.

Existen dos tipos de tiquete, estándar y premium. Ambos tienen un código compuesto por el número del vuelo y el número de silla, un vuelo asociado y un precio en euros. El premium, a diferencia del estándar, le permite elegir al usuario un combo de comida para disfrutar durante el vuelo y seleccionar una silla VIP en la aeronave.

En algunas ocasiones para los administradores resulta sencillo poder generar un archivo csv para cargar el programa de vuelos del mes, así que es importante poder cargar el programa de vuelos indicando la cantidad de archivos que se lograron cargar.

### **Cliente**

Los clientes al ingresar a la aplicación podrán crear su perfil y luego ingresar para ver la oferta de destinos que tiene el aeropuerto, además de buscar el



destino por nombre. Dentro de su perfil los usuarios podrán ver la lista de tiquetes comprados ordenados cronológicamente de forma descendente.

Desde el perfil los clientes podrán adquirir sus tiquetes, seleccionando su destino, la aerolínea de su preferencia que ofrezca vuelos al destino en el rango de fecha deseado por el usuario. Posteriormente, el usuario podrá seleccionar su silla, asignar el tipo de equipaje que lleva, enlazar su tiquete con un acompañante y seleccionar el tipo de tiquete que desea. En caso de seleccionar el tiquete premium se habilitará en la plataforma la selección del combo de comida que desee el usuario.

Sin importar el tipo de usuario todos podrán realizar actualizaciones a sus nombres de usuario, contraseña. Además, se necesita que cada nombre de usuario sea único.

## REQUERIMIENTOS FUNCIONALES DEL SISTEMA

El sistema debe estar en capacidad de:

**RQ1. Gestionar** los accesos a la información de cada usuario acorde a su perfil y tipo. Los clientes solo pueden ver su propia información de cliente. Adicional, los usuarios acordes a su dependencia pueden acceder solo a pantallas fijas basadas en su perfil.

**RQ1.1. Mostrar** un tablero único para el administrador del aeropuerto

**RQ1.1.1.** Permitir crear, actualizar y eliminar objetos de tipo supervisor de la torre de control

**RQ1.1.2.** Permitir crear, actualizar y eliminar objetos de tipo agente de migración

**RQ1.1.3.** Permitir crear, actualizar y eliminar objetos de tipo administrador de aerolínea

**RQ1.2. Mostrar** un tablero único para el supervisor de la torre de control

**RQ1.2.1.** Mostrar la lista de vuelos actuales

**RQ1.2.1.1.** Desplegar la información del vuelo, cuanto ha recorrido del vuelo, destino y el tiempo que lleva hasta el momento.

**RQ1.2.2.** Mostrar la lista de vuelos en crisis

**RQ1.2.3.** Mostrar el estado físico de las pistas de aterrizaje y de la torre de control

**RQ1.2.4.** Permitir parar la operación para realizar un mantenimiento preventivo

**RQ1.2.5.** Mostrar la lista de los próximos vuelos dentro de un periodo de 24 horas

**RQ1.2.6.** Mostrar la asignación de vuelos a pistas

**RQ1.3. Mostrar** un tablero único para el agente de migración

**RQ1.3.1.** Visualizar la información general del área de migración por medio de los indicadores solicitados.



**RQ1.3.2.** Generar reportes en csv de los indicadores de las ventas de tiquetes dentro de un rango de tiempo específico.

**RQ1.4. Mostrar** un tablero único para el administrador de aerolínea

**RQ1.4.1.** Completar el perfil de la aerolínea que asigna el administrador del aeropuerto, al administrador de la aerolínea

**RQ1.4.2.** Crear, actualizar y eliminar vuelos

**RQ1.4.3.** Cargar csv de vuelos

**RQ1.4.4.** Gestionar empleados de la aerolínea

**RQ1.4.4.1.** Crear, actualizar y borrar pilotos

**RQ1.4.4.2.** Crear, actualizar y borrar asistentes

**RQ1.4.4.3.** Cargar csv de pilotos y asistentes

**RQ1.5. Gestionar** los clientes (pasajeros). Cabe resaltar que todo cliente es una persona, por lo tanto, comparte el nombre y apellido, adicionando un correo, una contraseña y una posible lista enlazada de tickets comprados por el mismo.

**RQ1.5.1.** Permitir crear un nuevo usuario usando autenticación propia de java o (JWT).

**RQ1.5.1.1.** Asignar género dentro de configuración del perfil.

**RQ1.5.1.2.** Asignar edad dentro de configuración del perfil.

**RQ1.5.2.** Borrar su cuenta

**RQ1.5.3.** Visualizar la lista de tiquetes comprados

**RQ1.5.4.** Comprar nuevos tiquetes

**RQ1.5.4.1.** Buscar vuelos disponibles en los destinos

**RQ1.5.4.2.** Seleccionar la aerolínea

**RQ1.5.4.3.** Seleccionar la silla en el avión

**RQ1.5.4.4.** Seleccionar el tipo de equipaje

**RQ1.5.4.5.** Seleccionar el tipo de tiquete

**RQ1.5.4.6.** Asignar acompañante en caso de ser necesario

**RQ1.5.4.7.** Seleccionar el alimento en caso de estar disponible

**RQ1.5.4.8.** Generar un código de tiquete

**RQ1.5.5** Permitir crear un nuevo usuario (Cliente) usando autenticación OAuth2 propia del Api de Google.



- RQ2. Mostrar** permanentemente actualizada la fecha y hora actual en alguna parte de la ventana.
- RQ3. Garantizar** persistencia en la información.
- RQ4. Manejar** concurrencia en la zona de migración, es decir, cada que aterrice un vuelo se procederá a la correspondiente visualización de documentos por parte de los colaboradores del área migratoria del aeropuerto.
- RQ5. Identificar** en las zonas migratorias diferentes excepciones que conformarán una especie de barrera a los usuarios al momento de viajar.
- RQ5.1 Revisar** que se cuenten con los documentos pertinentes del menor que viaja sin sus padres.
- RQ5.2 Seleccionar** aquellas personas con antecedentes judiciales.
- RQ5.3 Revisar** los documentos para la tramitología necesaria para embarcar, esto debido a la emergencia sanitaria actual (Covid-19).
- RQ6. Proyectar** gráfica de barras con tres columnas que evidencian el paso de los pasajeros dentro de la migración, las cuales son: el número de personas que han pasado exitosamente, número de personas retenidas por el incumplimiento de algunas de las anteriores excepciones presentadas y el número de personas que faltan por atender.
- RQ7. Ordenar** las listas de los pilotos por medio de selección, los aviones por medio del algoritmo de inserción y la correspondiente implementación de orden en los reportes a exportar por la aerolínea.

## **JUSTIFICACIÓN**

Algunas aerolíneas no cuentan con un óptimo uso de los recursos, ni con una buena gestión de los procesos operacionales, por lo cual se ven obligadas a invertir su capital en gastos innecesarios. Adicional, por la falta de cumplimiento en sus trayectos crean un inconformismo en los usuarios.

El presente proyecto tiene como objetivo solucionar la gestión administrativa en organizaciones aeroportuarias con un software integral, sencillo e innovador. Por un lado, se enfocará en determinar las estrategias del sistema de operaciones, los procesos de mantenimiento y el análisis de las características físicas del sistema de compra y venta de tiquetes. Por otro lado, se realizará una investigación de aquellas variables que influyen en cada uno de los procesos y procedimientos en la cadena de los trabajos actualmente realizados en aeropuertos internacionales, lo cual se traducirá en la identificación de aquellas falencias o debilidades que se están presentando en cada etapa, todo ello enfocado a brindar posibles soluciones, para optimizar el proceso de ejecución del servicio prestado con calidad.

Para llevar a cabalidad el objetivo propuesto: en primer lugar, repartiremos funcionalidades específicas a cada miembro para mejorar tanto la productividad como eficacia en la resolución de cada uno de los requerimientos previamente mencionados. En segundo lugar, trabajaremos basados en un patrón de arquitectura de software conocido como MVC, que separa los datos y principalmente lo que es la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Por lo tanto, se hace indispensable contar con al menos tres integrantes dentro del proyecto, los cuales tendrán un rol de liderazgo dentro de cada paquete implementado (Modelo, Vista y Controlador).



## WIREFRAMES

Enlace: <https://internationalairportsystem.invisionapp.com/freehand/GUI-njfeBcvTe>

### Log In

< Logo >

Email Address

Password

Forgot password?

Log In

### User: Client

My Flights →

New Flight →

Log Out →

I A S

Tuesday, May 18, 2021  
15:21

Username

☒

Upcoming flights

Miami, US  
Madrid, SP

June 30, 2021  
17:00

Paris, FR  
New York, US

June 30, 2021  
17:00

Paris, FR  
New York, US

June 30, 2021  
17:00

New flight

### Tickets Screen

☐

☒

☐

☐

☐

I A S

Tuesday, May 18, 2021  
15:21

Username

Planning your next flight

Destination  
Enter a city

Dates  
Departure

Return

Search

Flights to <Destination> from <Departure>

Avianca

Departure BOG - NYC  
Wed, Sep 15, 2021  
17:00

From  
€200.00

Avianca

Departure BOG - NYC  
Wed, Sep 15, 2021  
12:00

From  
€185.00

## Choose your Seat

IAS

Tuesday, May 18, 2021  
15:21

Username

Departure BOG - NYC

Wed, Sep 15, 2021  
17:00

Ticket Type

▼

Your Seat | B9

Food (Only with Premium Ticket) ▼

Baggage ▼

Cancel

Buy Now

ID: 15A4B-9BSA1

A B C D E F

1

2

3

4

5

6

7

8

9

## User: Airport Administrator

Dashboard →

Control Tower Supervisors →

Migration Agents →

Airlines →

Log Out →

IAS

Tuesday, May 18, 2021  
15:21

Administrator

My airport

< icon > | Control Tower Supervisors

< icon > | Migration Agents

< icon > | Airlines

## Control Tower GUI

IAS

Tuesday, May 18, 2021  
15:21

Administrator

Control Tower Supervisors / Migration Agents

Search

Enter a name

Search

Add Supervisor / Agent

Name Identification Status

José Arango 1006741259 Active

Maria López 4210597850 Active

UNIVERSIDAD  
ICESI

## Actual and searching Airlines

IAS

Tuesday, May 18, 2021  
15:21

Administrator

☐  
  
☐  
  
☐  
  
☒  
  
☐

### Airlines

Name	Aircrafts	Tickets
< Icon > Avianca	20	1530
< Icon > American Airlines	12	895

## Airline Crud

IAS

Tuesday, May 18, 2021  
15:21

Administrator

☐  
  
☐  
  
☐  
  
☒  
  
☐

### New Airline

## User: Control Tower Supervisor

Active Flights →

Upcoming Flights →

Status →

New Maintenance →

Log Out →

☒  
  
☐  
  
☐  
  
☐  
  
☐

### Active Flights

Airline	Aircraft	Departure	Destination	Arrival	Elapsed	Status
Avianca	Airbus A330	Miami, US	Madrid, SP	18:34	01:25:14	Delayed
Avianca	Airbus A330	Miami, US	Madrid, SP	18:34	00:26:21	Delayed
Avianca	Airbus A330	Miami, US	Madrid, SP	18:34	00:08:01	Delayed

## Actual and searching flights

IAS

Tuesday, May 18, 2021  
15:21

Administrator

☐
☒
☐
☐
☐
☐

Upcoming Flights

Search

Enter an airline

Search

Airline	Aircraft	Departure	Destination	Arrival	Assigned Track
Avianca	Airbus A330	Miami, US	Madrid, SP	18:34	(Click to assign track)
Avianca	Airbus A330	Miami, US	Madrid, SP	18:34	Track01
Avianca	Airbus A330	Miami, US	Madrid, SP	18:34	(Click to assign track)

☐
☐

## Flight Status

IAS

Tuesday, May 18, 2021  
15:21

Administrator

☐
☐
☒
☐
☐
☐

Status

Control Tower

Flawless

Track

Status

Track01

Decayed

Track02

Flawless

Track03

Maintenance

## User: Airline Administrator

IAS

Tuesday, May 18, 2021  
15:21

Administrator

Airline Flights →

Employees →

Log Out →

☒
☐
☐
☐
☐

Airline Flights


Search

Enter a flight

Search

Aircraft	Departure	Destination	Date	Tickets
Airbus A330	Miami, US	Madrid, SP	18:34	54/150
Airbus A330	Miami, US	Madrid, SP	18:34	150/150
Airbus A330	Miami, US	Madrid, SP	18:34	20/150

☐
☐
☐



UNIVERSIDAD  
ICESI

## Airline Employees

I A S

Tuesday, May 18, 2021  
15:21

Administrator

Airline Employees

Search

Enter a name

Search

Add Supervisor / Agent

Name	Identification	Status	Juan S.
José Arango	1006741259	Active	<div><div></div><div>&gt;</div></div>
Maria López	4210597850	Active	<div><div></div><div>&gt;</div></div>

## Employee Crud

**I A S**

Tuesday, May 16, 2020  
15:21 Administrator

### New Employee

First Name	Last Name
Identification Number	
Employee Type ▼	
Cancel	Add Employee

**User:** Migration Agent

[Enlace documento original - Acercar y ver los diagramas](#)

## Presentación pruebas unitarias JUnit


A continuación se presenta el desarrollo a realizar de las pruebas unitarias de los métodos del sistema para el nuevo aeropuerto internacional de Madrid, se presentan los escenarios y los casos. Con el propósito de facilitar el entendimiento de la aplicación se presentan las pruebas a realizar en cada uno de los contextos marcados por el tipo de usuario que usa la aplicación.

## Administrador General Aeropuerto - Casos

### Crear nuevo usuario administrador del aeropuerto

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en un solo caso, añadir correctamente el objeto la relación está null.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario1	Airport	

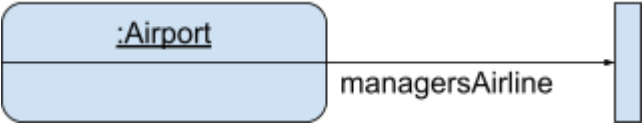
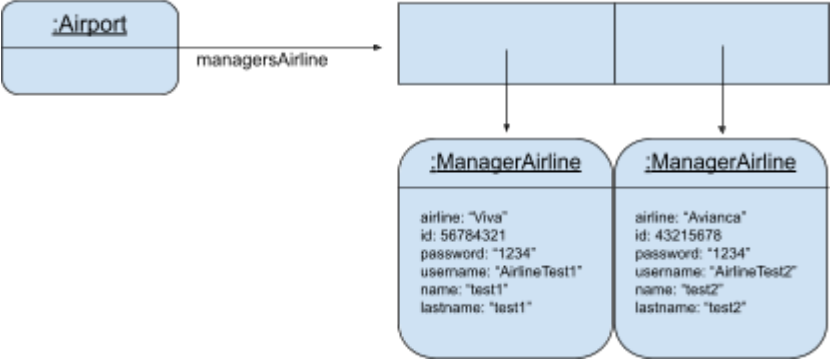
### Diseño de Casos de Prueba

Objetivo de la Prueba: Registrar un nuevo usuario administrador general.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	createMainUser	setupScenario1	username:"general IMG" name: "test" lastname: "testL" password:"1234"	User created successfully. Returns the name of the main user and the ID

## Crear nuevo usuario administrador de aerolínea

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

## Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario2	Airport	 <pre> sequenceDiagram     participant Airport as :Airport     Airport-&gt;&gt;[] managersAirline           </pre>
setupScenario3	Airport	 <pre> sequenceDiagram     participant Airport as :Airport     participant ManagerAirline1 as :ManagerAirline     participant ManagerAirline2 as :ManagerAirline     Airport-&gt;&gt;[] managersAirline     []--&gt;&gt;ManagerAirline1     []--&gt;&gt;ManagerAirline2     ManagerAirline1-&gt;&gt;ManagerAirline2           </pre>

## Diseño de Casos de Prueba

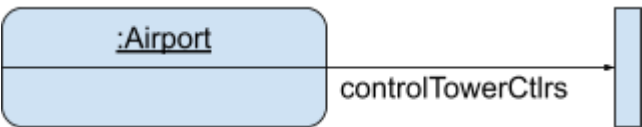
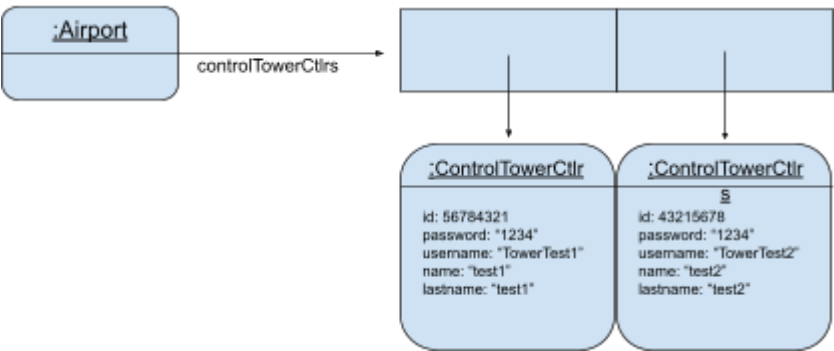
Objetivo de la Prueba: Registrar un nuevo usuario administrador de aerolínea				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	createAirlineManager	setupScenario2	airline: "Copa" id: 12345678 password: "1234" username: "AirlineTest" name: "test" lastname: "testL"	User created successfully. Returns Copa, 12345678, 1234, AirlineTest, test, test.

Airport	createAirlineManager	setupScenario3	airline: "Copa" id: 12345678 password: "1234" username: "AirlineTest" name: "test" lastname: "testL"	User created successfully. Returns Copa, 12345678, 1234, AirlineTest, test, test.
---------	----------------------	----------------	--	--

### Crear nuevo usuario controlador torre de control

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario4	Airport	
setupScenario5	Airport	




## Diseño de Casos de Prueba

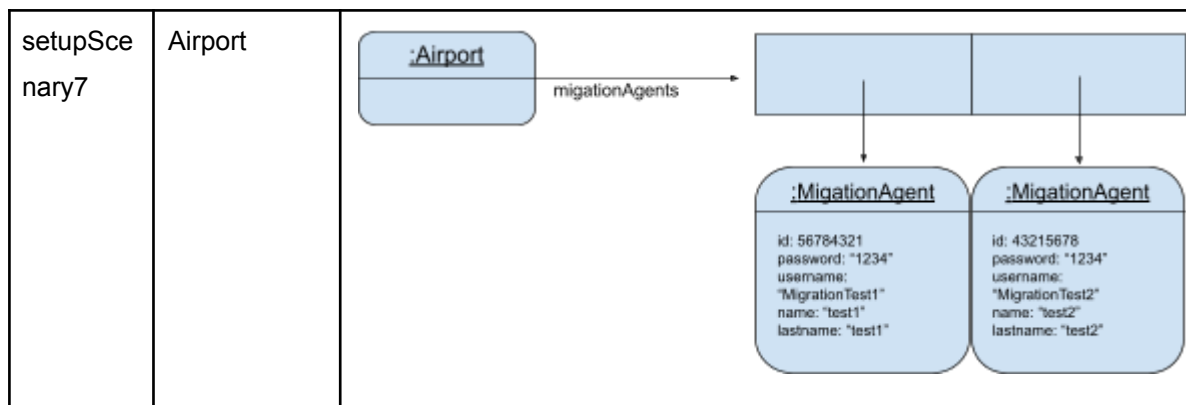
Objetivo de la Prueba: Registrar un nuevo controlador torre de control				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	createTowerCtrl	setupScenario4	id: 12345 password: "1234" username: "TowerTest" name: "test" lastname: "testL"	User created successfully. Returns 12345, 1234, TowerTest, test, testL.
Airport	createTowerCtrl	setupScenario5	id: 12345 password: "1234" username: "TowerTest" name: "test" lastname: "testL"	User created successfully. Returns 12345, 1234, TowerTest, test, testL.

### Crear nuevo usuario agente de migración

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario6	Airport	



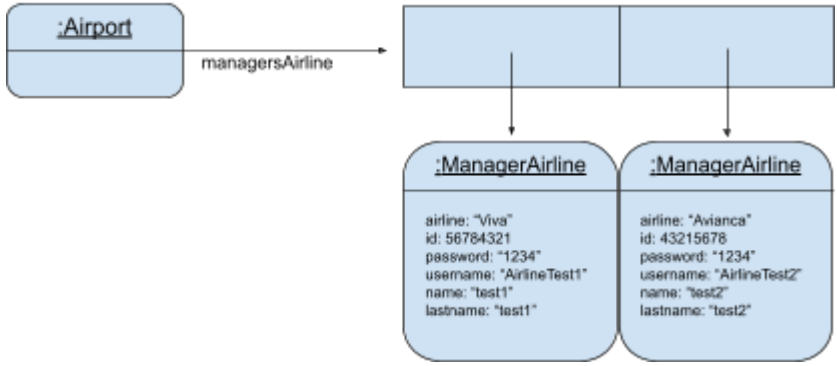
## Diseño de Casos de Prueba

Objetivo de la Prueba: Registrar un nuevo agente de migración				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	createM igration Agent	setupScen ary6	id: 12345 password: "1234" username: "MigrationTest1" name: "test" lastname: "testL"	User created successfully. Returns 12345, 1234, MigrationTest1, test, testL.
Airport	createM igration Agent	setupScen ary7	id: 12345 password: "1234" username: "MigrationTest1" name: "test" lastname: "testL"	User created successfully. Returns 12345, 1234, MigrationTest1, test, testL.

### Eliminar usuario administrador de aerolínea

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenary3	Airport	

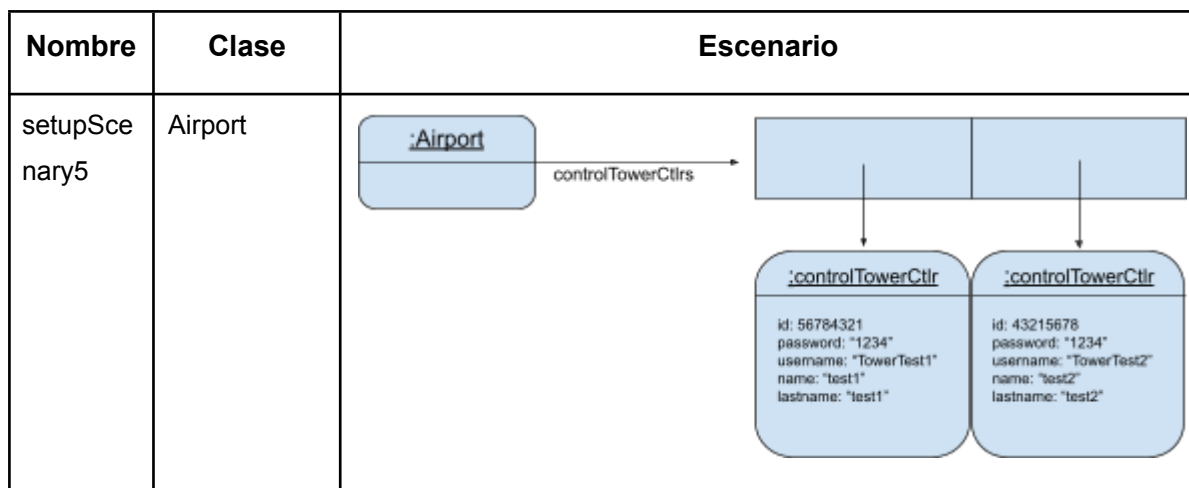
### Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar administrador de aerolínea por Id.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	deletAirlineManager	setupScenary3	id: 56784321	User eliminated successfully. Returns array size = 1.

### Eliminar usuario controlador torre de control

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios



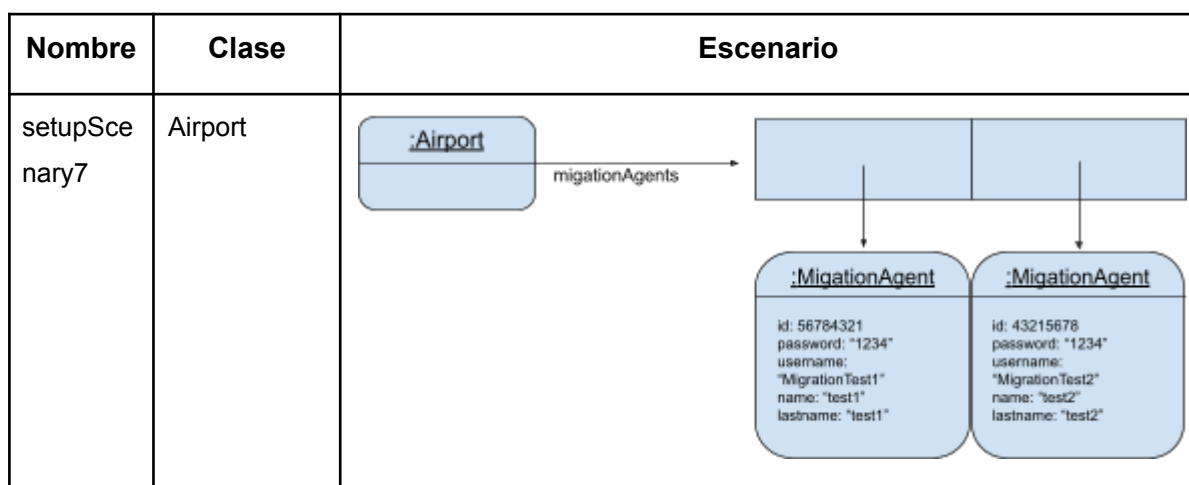
## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar controlador de torre por id				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	deleteTowerCtrl	setupScenario5	id: 56784321	User eliminated successfully. Returns array size = 1.

## Eliminar usuario agente de migración

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios



## Diseño de Casos de Prueba

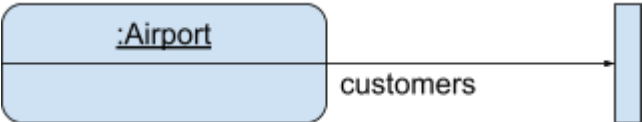
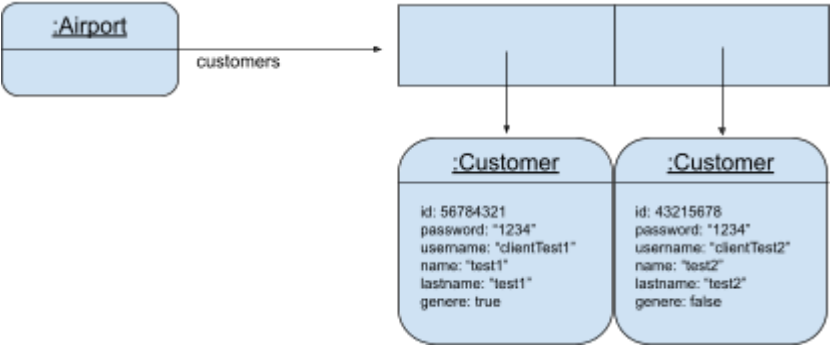
Objetivo de la Prueba: Eliminar agente de migración por id				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	deleteMigrationAgent	setupScenario7	id: 56784321	User eliminated successfully. Returns array size = 1.

## Cliente Final - Casos

### Crear nuevo usuario cliente

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario8	Airport	 <pre> sequenceDiagram     participant Airport as :Airport     Airport--&gt;&gt; customers: customers     </pre>
setupScenario9	Airport	 <pre> sequenceDiagram     participant Airport as :Airport     Airport--&gt;&gt; customers: customers     customers--&gt;&gt; Customer1:      customers--&gt;&gt; Customer2:      Customer1--&gt;&gt; Airport: id: 56784321, password: "1234", username: "clientTest1", name: "test1", lastname: "test1", genere: true     Customer2--&gt;&gt; Airport: id: 43215678, password: "1234", username: "clientTest2", name: "test2", lastname: "test2", genere: false     </pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Registrar un nuevo cliente.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	create Client	setupScen ary8	id: 12345 password: "1234" username: "clientTest" name: "test" lastname: "testL" genere: true	User created successfully. Returns 12345, 1234, clientTest, test, testL.
Airport	create Client	setupScen ary9	id: 12345 password: "1234" username: "clientTest" name: "test" lastname: "testL" genere: true	User created successfully. Returns 12345, 1234, clientTest, test, testL.

## Eliminar usuario usuario cliente

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios

Nombre	Clase	Escenario
setupScen ary9	Airport	<pre> classDiagram     class Airport {         +customers     }     class Customer {         +id         +password         +username         +name         +lastname         +genere     }     Airport --&gt; "2" Customer : customers         </pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar cliente por Id.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	delete Client	setupScenario9	id: 56784321	User eliminated successfully. Returns array size = 1.

### Crear nuevo ticket

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario1	Client	<pre> sequenceDiagram     participant Client as :Client     Client-&gt;&gt;[] : rootTicket           </pre>
setupScenario2	Client	<pre> sequenceDiagram     participant Client as :Client     Client-&gt;&gt;[] : rootTicket     activate []     []--&gt;&gt;Ticket1 as :Ticket     activate Ticket1     Ticket1--&gt;&gt;Ticket2 as :Ticket     activate Ticket2     Ticket2--&gt;&gt;[] :      deactivate Ticket2     deactivate Ticket1     deactivate []           </pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Crea un nuevo tiquete.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Client	createTicket	setupScenario1	id: 12345 to: "China" date: "8/5/2021 10:00 pm" type: "premium" seat: "1A" companion: false	Ticket created successfully. Returns 12345, China, 8/5/2021 10:00 pm, premium, 1A, false.
Client	createTicket	setupScenario2	id: 12345 to: "China" date: "8/5/2021 10:00 pm" type: "premium" seat: "1A" companion: false	Ticket created successfully. Returns 12345, China, 8/5/2021 10:00 pm, premium, 1A, false.

## Eliminar ticket

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario2	Client	<pre> graph LR     Client1[":Client"] -- tickets --&gt; Container     subgraph Container         Ticket[":Ticket&lt;br/&gt;id: 12345&lt;br/&gt;to: 'Colombia'&lt;br/&gt;date: '6/15/2021 5:00 pm'&lt;br/&gt;type: 'standard'&lt;br/&gt;seat: '15C'&lt;br/&gt;companion: false"]         Client2[":Client&lt;br/&gt;id: 54321&lt;br/&gt;to: 'Austria'&lt;br/&gt;date: '7/25/2021 1:00 pm'&lt;br/&gt;type: 'premium'&lt;br/&gt;seat: '2A'&lt;br/&gt;companion: false"]     end </pre>

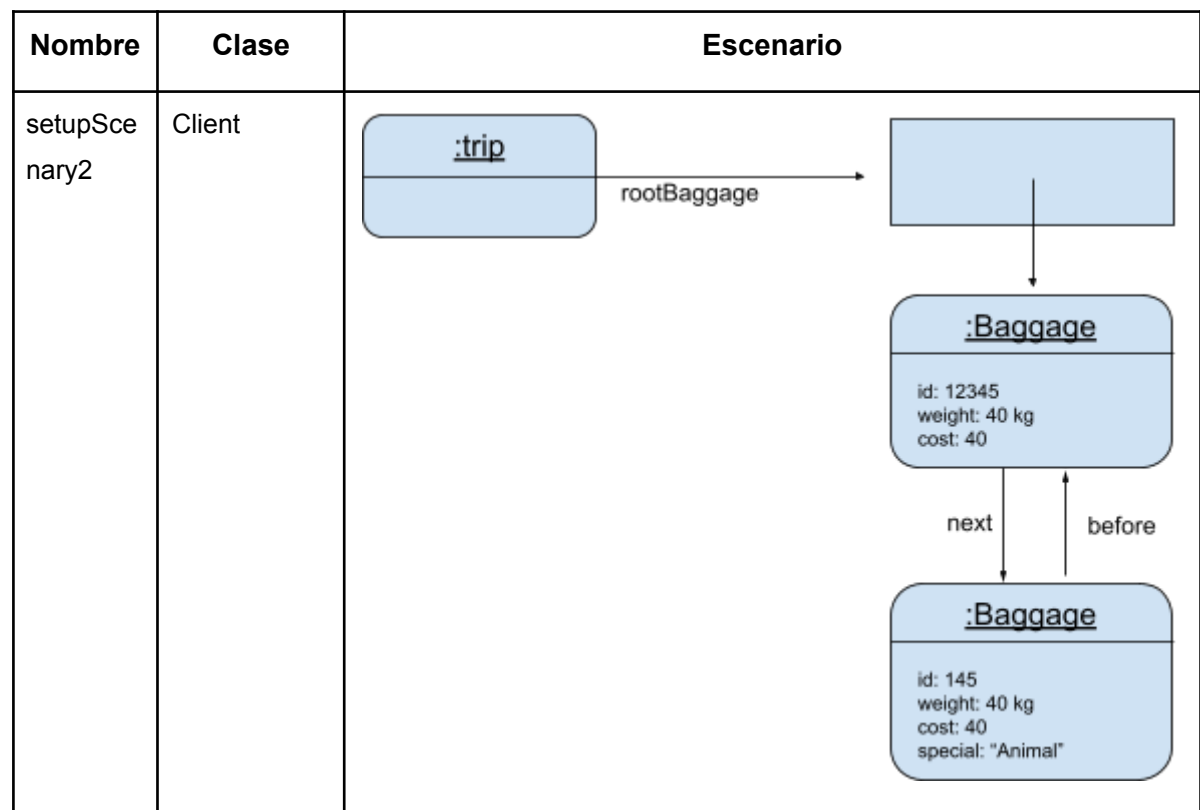


## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar tickete por Id.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Client	deletetic ket	setupScen ary11	id: 54321	Ticket eliminated successfully. Returns array size = 1.

## Añadir equipaje

### Configuración de los Escenarios



## Diseño de Casos de Prueba

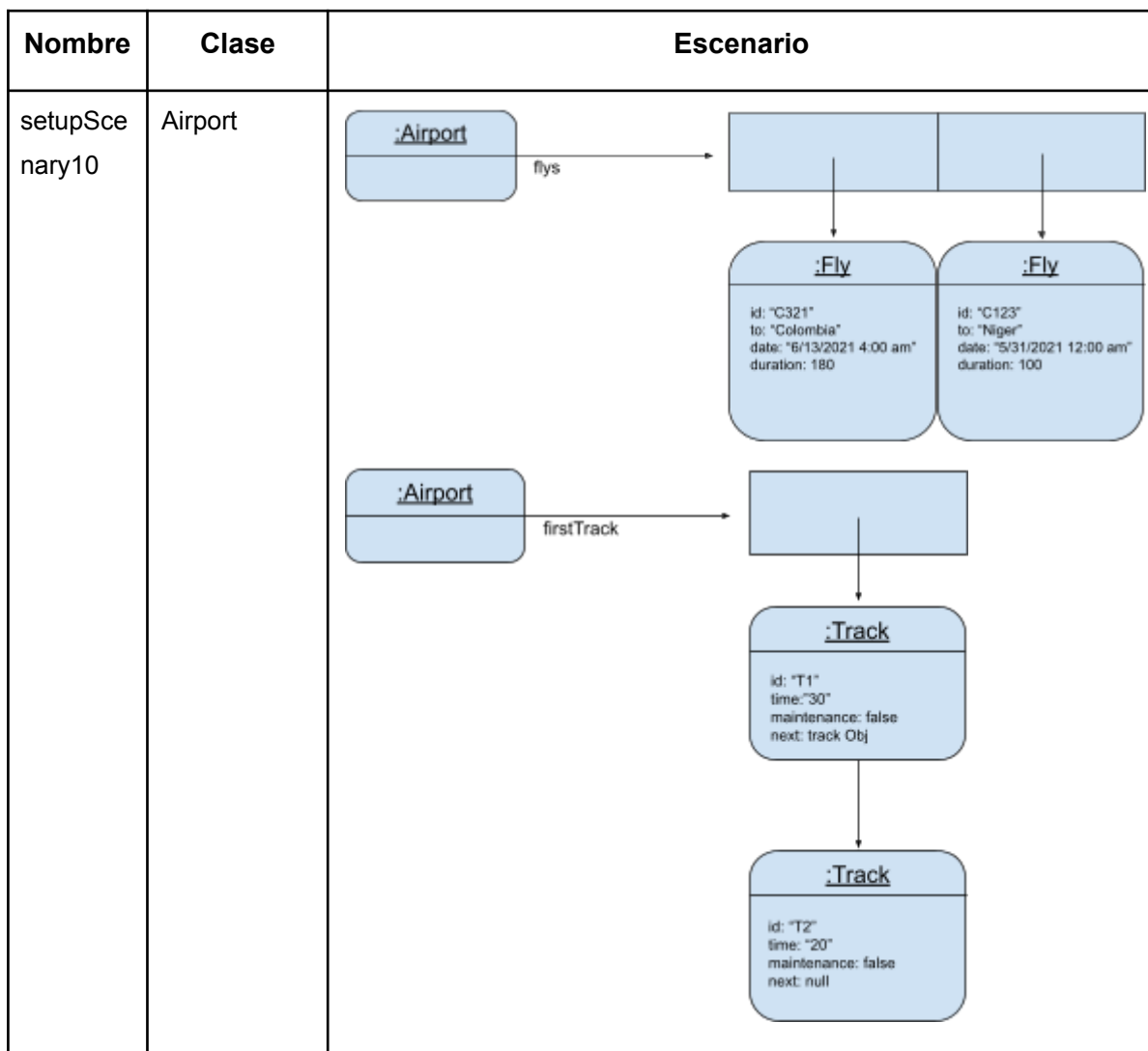
Objetivo de la Prueba: Añadir equipaje al vuelo				
Clase	Método	Escenario	Valores de Entrada	Resultado
Client	deletetic ket	setupScen ary11	id: 100 weight: 1	Baggage added successfully. Returns 100, 1, 100, gun.

			cost: 100 special: "gun"	
--	--	--	-----------------------------	--

## Controlador torre de control - Casos

### Asignar vuelo a pista

### Configuración de los Escenarios

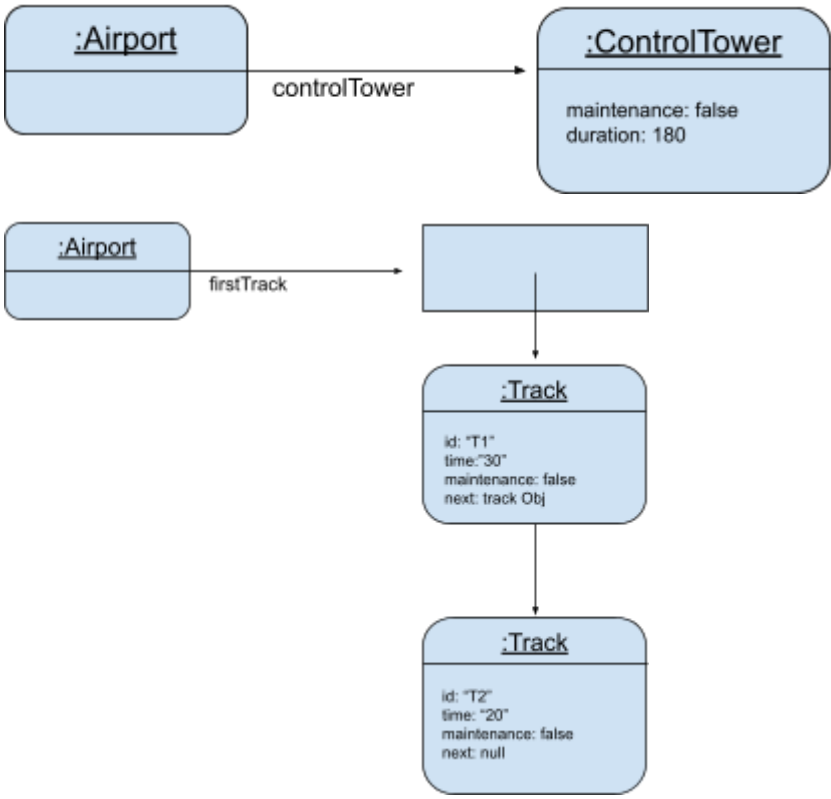


Diseño de Casos de Prueba

Objetivo de la Prueba: Asignar vuelo y una pista por un tiempo				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	assingFlyTrack	setupScenario10	id: C321 id: T1	true

Activar mantenimiento

Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario11	Airport	 <pre>graph TD     Airport1[":Airport"] -- controlTower --&gt; ControlTower[":ControlTower&lt;br/&gt;maintenance: false&lt;br/&gt;duration: 180"]     Airport2[":Airport"] -- firstTrack --&gt; Obj1[" "]     Obj1 --&gt; Track1[":Track&lt;br/&gt;id: 'T1'&lt;br/&gt;time: '30'&lt;br/&gt;maintenance: false&lt;br/&gt;next: track Obj"]     Track1 --&gt; Track2[":Track&lt;br/&gt;id: 'T2'&lt;br/&gt;time: '20'&lt;br/&gt;maintenance: false&lt;br/&gt;next: null"]</pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Activar el período de mantenimiento				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	mainten ance	setupScen ary11		mantinance: true

## Llamar lista de vuelos actuales

## Configuración de los Escenarios

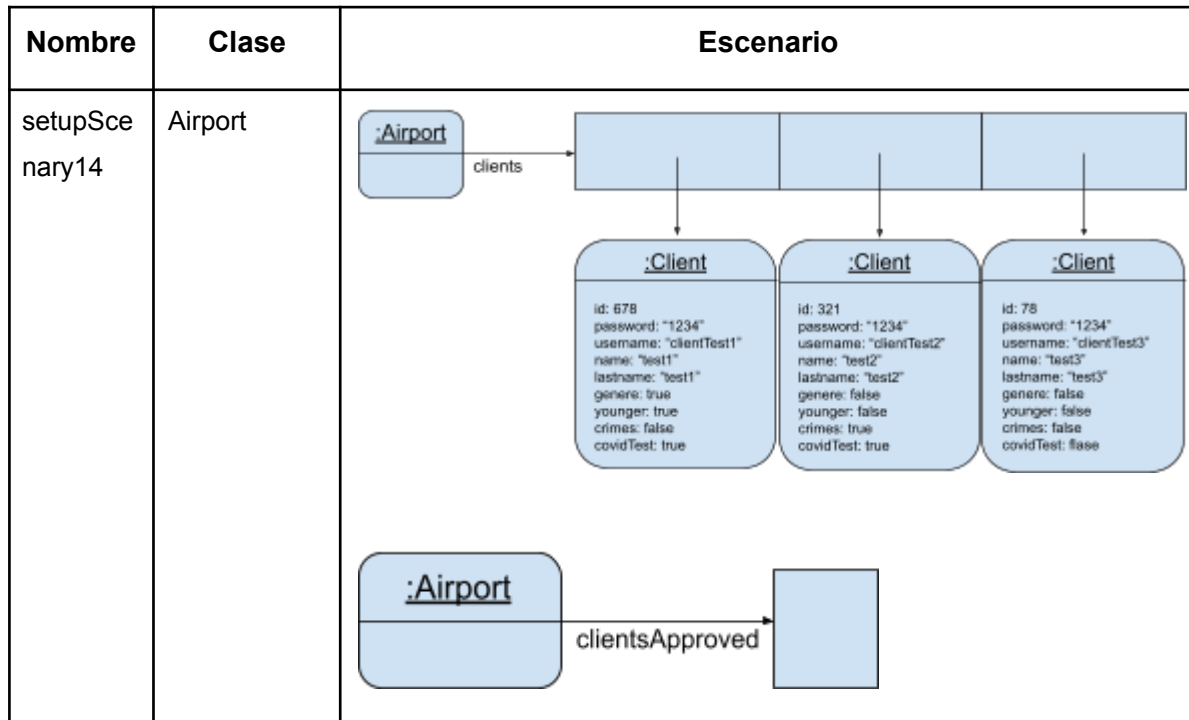
Nombre	Clase	Escenario
setupScen ary13	Airport	<pre>graph LR     Airport[":Airport"] -- flys --&gt; Fly1[":Fly&lt;br/&gt;id: 'C321'&lt;br/&gt;to: 'Colombia'&lt;br/&gt;date: '8/13/2021 4:00 am'&lt;br/&gt;duration: 180"]     Airport -- flys --&gt; Fly2[":Fly&lt;br/&gt;id: 'C123'&lt;br/&gt;to: 'Niger'&lt;br/&gt;date: '5/31/2021 12:00 am'&lt;br/&gt;duration: 100"]</pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Obtener lista de vuelos				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	getActu alFlys	setupScen ary13		id: C321 id: C123

## Agente de migración - Casos

### Aprobación de migración para pasar Configuración de los Escenarios



### Diseño de Casos de Prueba

Objetivo de la Prueba: Validar la salida de un cliente				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	validateClientPass	setupScenario14	id: 678	fail clientsApproved size = 0
Airport	validateClientPass	setupScenario14	id: 321	fail clientsApproved size = 0


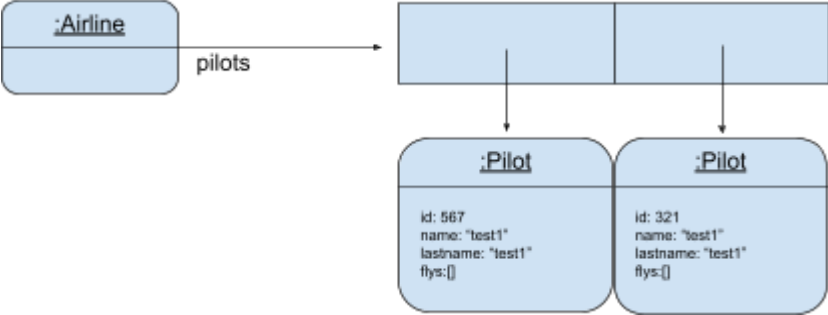
Airport	validat eClient Pass	setupScen ary14	id: 78	fail clientsApproved size = 0
---------	----------------------------	--------------------	--------	----------------------------------

## Administrador de aerolínea - Casos

### Crear nuevo piloto

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenary1	Airline	 <pre> sequenceDiagram     participant A as :Airline     participant B as      A-&gt;&gt;B: pilots   </pre>
setupScenary2	Airline	 <pre> sequenceDiagram     participant A as :Airline     participant B as      participant C as :Pilot     participant D as :Pilot     A-&gt;&gt;B: pilots     B-&gt;&gt;C:      B-&gt;&gt;D:      Note over C: id: 567 name: "test1" lastname: "test1" flys: []     Note over D: id: 321 name: "test1" lastname: "test1" flys: []   </pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Crear un nuevo piloto de la aerolínea				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	create Pilot	setupScenary1	id: 123 name: "test" lastname: "testL" flys:[]	Created successfully. Returns 123, test, testL, [].
Airline	create Pilot	setupScenary2	id: 123 name: "test" lastname: "testL" flys:[]	Created successfully. Returns 123, test, testL, [].

### Crear nuevo asistente

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenary3	Airline	<pre> sequenceDiagram     participant Airline as :Airline     Airline--&gt;&gt; assistants: assistants     </pre>
setupScenary4	Airline	<pre> sequenceDiagram     participant Airline as :Airline     participant Container     Airline--&gt;&gt; Container: assistants     Container--&gt;&gt; Assistant1 as :Assistant     Container--&gt;&gt; Assistant2 as :Assistant     Assistant1--&gt;&gt; details1: id: 567, name: 'test1', lastname: 'test1', flys: []     Assistant2--&gt;&gt; details2: id: 321, name: 'test1', lastname: 'test1', flys: []     </pre>

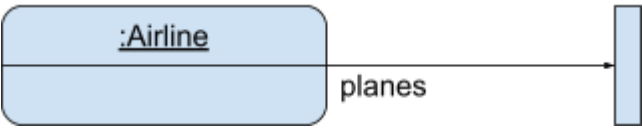
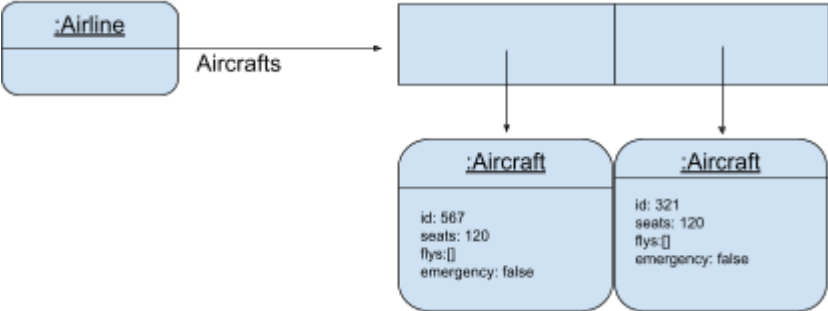
## Diseño de Casos de Prueba

Objetivo de la Prueba: Crear un nuevo asistente de vuelo				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	createAssistant	setupScenario3	id: 123 name: "test" lastname: "testL" flys:[]	Created successfully. Returns 123, test, testL, [].
Airline	createAssistant	setupScenario4	id: 123 name: "test" lastname: "testL" flys:[]	Created successfully. Returns 123, test, testL, [].

### Crear nuevo avión

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario5	Airline	 <pre> classDiagram     class Airline {     }     Airline --&gt; [] planes           </pre>
setupScenario6	Airline	 <pre> classDiagram     class Airline {     }     class Aircraft {         id: 567         seats: 120         flys: []         emergency: false     }     class Aircraft2 {         id: 321         seats: 120         flys: []         emergency: false     }     Airline --&gt; [ ] Aircrafts     Aircrafts --&gt; Aircraft     Aircrafts --&gt; Aircraft2           </pre>



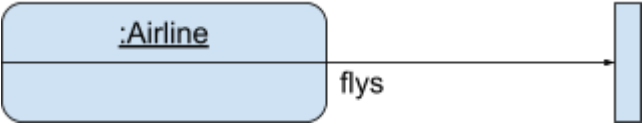
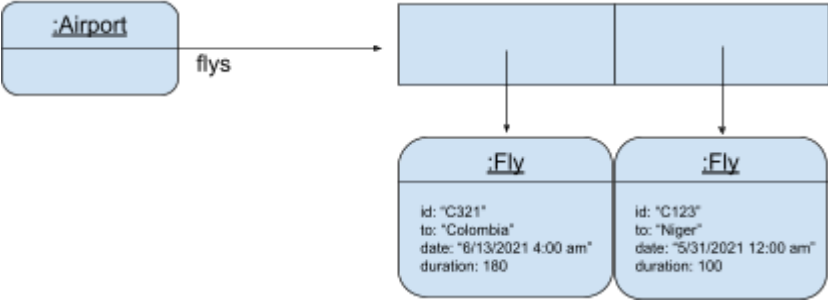
## Diseño de Casos de Prueba

Objetivo de la Prueba: crear un nuevo avión				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	createPlane	setupScenario5	id: 123 seats: 150 flights: [] emergency: false	Created successfully. Returns 123, 150, [], false.
Airline	createPlane	setupScenario6	id: 123 seats: 150 flights: [] emergency: false	Created successfully. Returns 123, 150, [], false.

### Crear nuevo vuelo

Se presenta a continuación el método encargado de **añadir una nueva instancia**, en 2 casos, añadir correctamente el objeto al arreglo está vacío y cuando hay otros objetos en el.

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario7	Airline	
setupScenario8	Airline	

## Diseño de Casos de Prueba

Objetivo de la Prueba: Crear un nuevo registro de vuelo en la aerolínea				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	createFly	setupScenario7	id: "C452" to: "Argentina" date: "8/2/2021 9:00 pm" duration: 280	Created successfully. Returns C452, Argentina, 8/2/2021 9:00 pm, 280.
Airline	createFly	setupScenario8	id: "C452" to: "Argentina" date: "8/2/2021 9:00 pm" duration: 280	Created successfully. Returns C452, Argentina, 8/2/2021 9:00 pm, 280.

## Eliminar piloto

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario2	Airline	<pre> classDiagram     class Airport {         +pilots     }     class Pilot {         +id         +name         +lastname         +flys     }     Airport --&gt; Pilot : pilots     </pre>

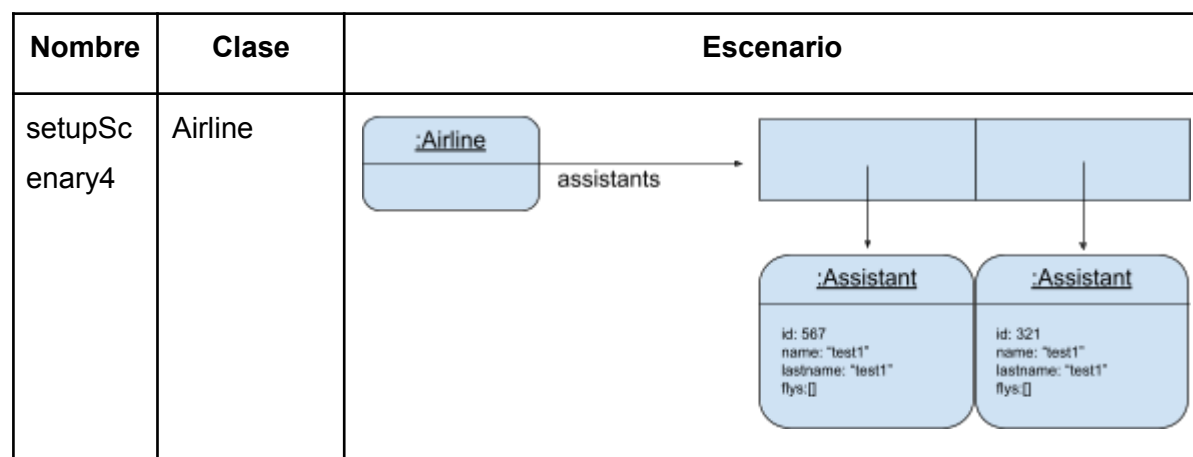
## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar un piloto por id				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	deletPlane	setupScenario2	id: 567	User eliminated successfully. Returns array size = 1.

### Eliminar asistente

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

### Configuración de los Escenarios



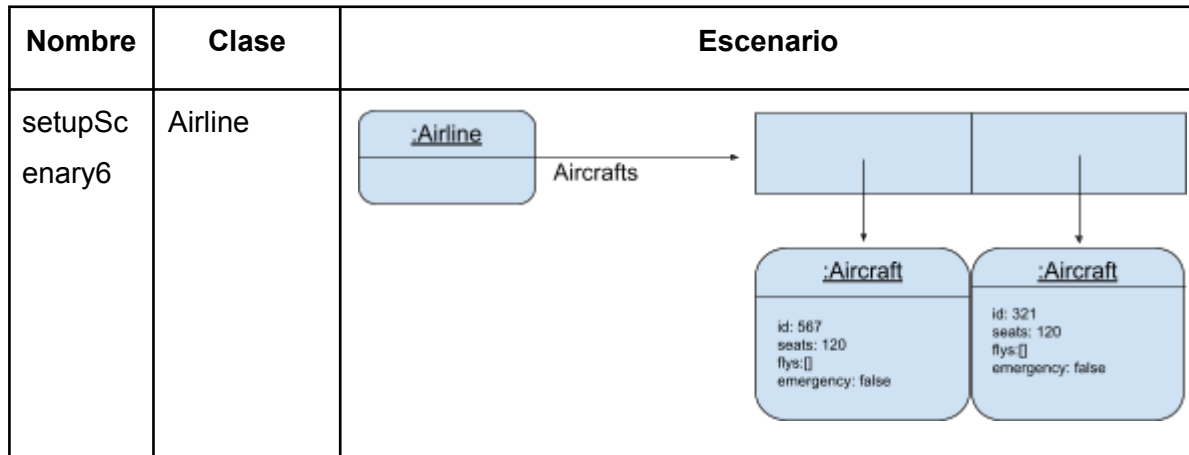
## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar asistente de vuelo por id				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	deletAssistant	setupScenario4	id: 321	User eliminated successfully. Returns array size = 1.

### Eliminar avión

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios



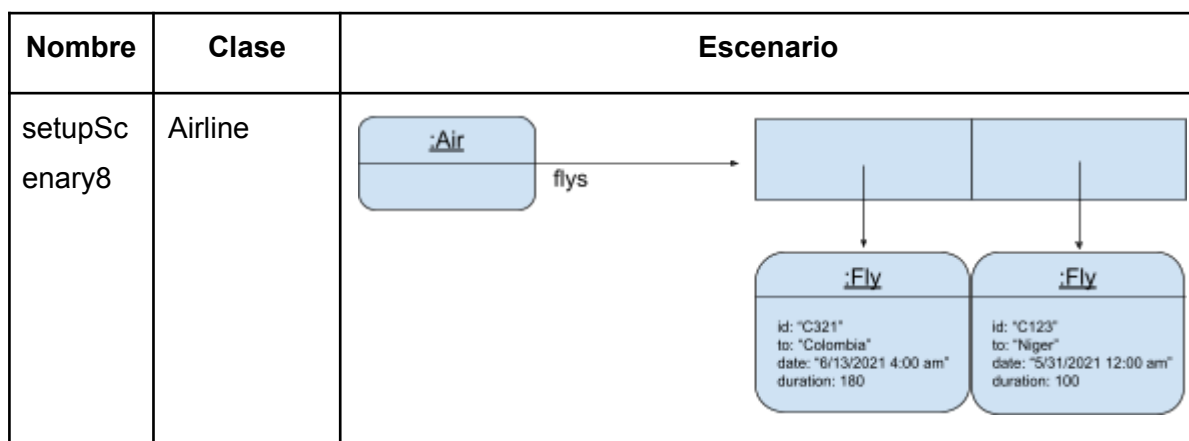
## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar un avión por id				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	deletePlane	setupScenario6	id: 567	User eliminated successfully. Returns array size = 1.

## Eliminar vuelo

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios



## Diseño de Casos de Prueba

Objetivo de la Prueba: Eliminar un vuelo por id				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	deleteFlight	setupScenario8	id: "C123"	User eliminated successfully. Returns array size = 1.

## Asignar Asistentes a vuelo

Se presenta a continuación el método encargado de **eliminar una instancia**, en un caso, eliminar cuando hay otros objetos en el arreglo de instancias.

## Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenario9	Airline	<pre>graph TD     A["&lt;u&gt;:Aircraft&lt;/u&gt;"] -- rootAssitent --&gt; B["&lt;u&gt;:Aircraft&lt;/u&gt;"]     B --&gt; C["&lt;u&gt;:Aircraft&lt;/u&gt;"]     A --- A_attr["id: 567&lt;br/&gt;name: 'test1'&lt;br/&gt;lastname: 'test1'&lt;br/&gt;flies: []"]     C --- C_attr["id: 321&lt;br/&gt;name: 'test2'&lt;br/&gt;lastname: 'test2'&lt;br/&gt;flies: []"]</pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Añadir un asistente al vuelo				
Clase	Método	Escenario	Valores de Entrada	Resultado
Airline	deletPlane	setupScenary9	id: 678 name: "test" lastname: "testL" flys:[]	Assistant added successfully. Returns 678, test, testL, [].

