Australis Airlines

Introducción a la Programación II
Trabajo Práctico Especial
2do Cuatrimestre 2017

Objetivo

Se desea implementar un sistema de gestión de reservas y venta de tickets para una aerolínea. El sistema a realizar para cada reserva debe controlar la disponibilidad de los tickets para cada pasajero y para cada escala a realizar.

Se requerirá el desarrollo de 3 aplicaciones con interfaz gráfica de consola:

1. Aplicación para empleados de la aerolínea que realizan ventas y configuran las entidades.
2. Aplicación para clientes que realizan compras y quieren ver sus reservas.
3. Aplicación para los pilotos y oficiales de abordo, para consultar su itinerario de viajes.

Además será requerido construir una aplicación que sea el API del sistema.

Enunciado

La aerolínea tiene empleados de tipos diferentes, sean administrativos, pilotos, y personal de abordo.
Los administrativos pertenecen a una única área, y la misma determina si está habilitada para realizar una venta o no. Solamente algunas áreas pueden hacer venta de pasajes.

Un vuelo siempre es entre dos aeropuertos conocidos y están planificados para un dia de la semana y horario en particular, y tienen un código. Además tienen una fecha desde y hasta que determinan un rango de semanas en donde ese vuelo se repite, conservando siempre el mismo código de vuelo. Un vuelo también tiene vinculado un avión, donde cada avión tiene un código único y un mapa de asientos y un tarifario de precios por categoría de asientos. Hay distintas categorías de asientos, por lo que las categorías deben ser dinámicas.

Cuando se vende un pasaje, el mismo puede ser para una o más personas, y puede ser solamente ida o ida y vuelta, y el usuario puede querer viajar en alguna categoría en particular o no, y por último puede querer 0 escalas o un máximo de escalas, siendo siempre el máximo del sistema 3 escalas, es decir 4 vuelos para una reserva.

En la compra de pasajes se deberá contar con un formulario para ingreso de datos. Este es el paso 1 de la compra. Los datos del formulario serán:

1. Aeropuerto de origen y destino
2. Fechas de salida y llegada
3. Cantidad de pasajeros
4. Cantidad de escalas máxima (pudiendo seleccionar que sea indistinta)
5. Categoría deseada (pudiendo seleccionar que sea indistinta)
6. Selección de criterio de orden para los resultados (siendo los mínimos requeridos: precio, escalas y duración).

Luego, como paso 2 se deben mostrar los resultados pertinentes con un detalle de cada oferta, y solicitar al usuario si desea comprar una de dichas ofertas o no.
Al momento de comprar, y como paso 3, el API deberá validar el pago contra un Webservice provisto por la cátedra, que hará las veces de gateway de pago.
Estos 3 pasos aplican tanto para la app de usuario final y la de vendedores de aerolínea.

Siempre para una reserva se deben generar tickets por cada persona y vuelo, y cada ticket debe contar con datos del dni del pasajero, asiento, vuelo, y fecha. Además el ticket debe tener un código único.

Aclaraciones:

1. Las diferentes aplicaciones deberán comunicarse por webservices con el API. Para esto se usará el framework Jersey para exponer el API y Retrofit para consumir el API desde los clientes.
2. Existirán ciertos métodos del API que deberán respetar un contrato definido por la cátedra en el Anexo 1.
3. La solución deberá tener al menos un test para cada proyecto en junit.
4. No se consideran impuestos ni ninguna modificación sobre los precios a fines de simplificar el enunciado.
5. Todas las compras deben quedar guardadas en un archivo, que debe ser cargado cada vez que se inicia el API, para no vender los mismos vuelos.

Plagio

Se pueden consultar ideas y estrategias de implementación con otros grupos, pero no se puede copiar el código. El plagio entre grupos será penalizado con la desaprobación del trabajo práctico de cada grupo involucrado.

Entregables

El material a entregar se detalla a continuación:

- Informe que detalle el diseño elegido y su justificación.
- Versión digital del código fuente
- Script de compilación para generar el ejecutable (jar) y su ejecución

Plan de trabajo

Se harán 3 entregas a modo de demo, para validar desarrollo, concepto y evolución del sistema.
Los grupos deberán ser de 3 o 4 personas.
Cada entrega tendrá una calificación que será utilizada para la nota final del TP.

La nota del TP será de la siguiente manera: 20% entrega 1 + 30% entrega 2 + 50% entrega final.

La fecha de inicio es el 30/09/2017.
Conformación de los grupos, fecha límite, el 06/10/2017.
La entrega 1 será el 13/10/2017.
La entrega 2 será el 27/10/2017.
La entrega 3 (final) será el 10/11/2017.

Consideraciones Técnicas

Deberá usarse un repositorio git propio de cada grupo y se deberá dar acceso a la cátedra.
Deberán realizarse tests con JUnit 4.
Deberán cumplirse las pautas indicadas oportunamente por la cátedra para poder vincular el proyecto al sistema de integración continua asignado.

Anexo I

API Server Gestión de Reservas (Alumnos)

Los alumnos deberán exponer el siguiente webservice:

```
Url: /flights/buy
Method: POST
Body:
{
      'client_id':1,
      'items': [
            {
                  'person_dni':33555888,
                  'flight_code':'AU002255',
                  'date':'2017-12-28',
                  'seat_code':'15A'
            }
            ,
            {
                  'person_dni':335554448,
                  'flight_code':'AU002255',
                  'date':'2017-12-28',
                  'seat_code':'15B'
            }
            ,
```

```
            ...
      ]
}

Response si hay Error:
{
      'result':'error',
      'exception':'PagoInvalidoException'
      'message':'...'
}

Response si se vende OK:
{
      'result':'ok',
      'booking_id':'55779',
}
```

API Server Gateway de Pagos (Cátedra)

La cátedra proveerá un API de ejemplo para el gateway de pago, que no deberá ser modificado, ya que para realizar la evaluación la cátedra podrá ejecutar el API server del alumnado contra un Gateway de Pago de Prueba de la cátedra.

La cátedra también proveerá el ejemplo con Retrofit para consumir el Gateway de Pago.