Memoria práctica ISD 2018/2019

Alejandro Romero Rivera

Laura Iglesias Gondar

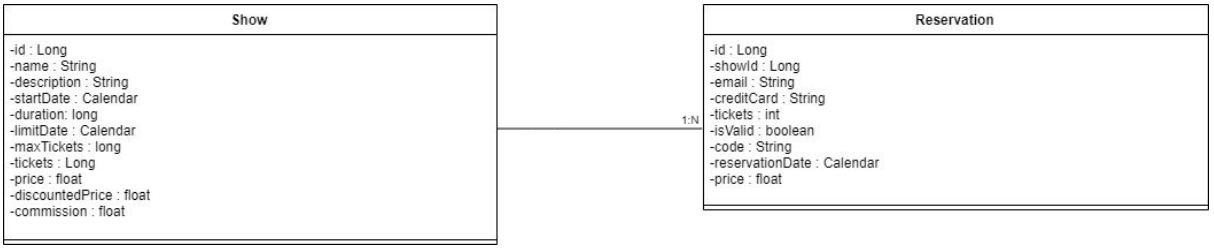
1. Introducción

Desarrollo de un servicio y clientes para la gestión de venta de entradas de espectáculos mediante la tecnología REST y SOAP.

De momento no se ha realizado ningún trabajo tutelado.

2. Capa modelo

2.1. Entidades



* El número de entradas restantes es guardado en la entidad Show aunque pueda ser un atributo calculado, por motivos de rendimiento.

2.2. DAOs

SqlReservationDao
+create(c : Connection, reservation : Reservation) : Reservation +update(c : Connection, reservation : Reservation) : void +remove(c : Connection, id : Long) : void +findByEmail(c : Connection, email : String) : List<Reservation> +findByCode(c : Connection, code : String) : Reservation

SqlShowDao
+create(c : Connection, show : Show) : Show +update(c : Connection, show : Show) : void +remove(c : Connection, id : Long) : void +find(c : Connection, id : Long) : Show +find(c : Connection, words : String, startDate : Calendar, endDate : Calendar) : List<Show>

2.3. Fachadas

TicketSellerService
+createShow(show : Show) : Show +updateShow(show : Show) : void +findShow(id : Long) : Show +findShows(keywords: String, start : Calendar, end : Calendar) : List<Show> +buyTickets(showId : Long, email : String, cardNumber : String, count : int): Reservation +getUserReservations(email : String) : List<Reservation> +checkReservation(code : String, cardNumber : String) : void

3. Capa de servicios

3.1. DTOs

ServiceReservationDto
-id : Long -showId : Long -email : String -creditCard : String -tickets : int -isValid : boolean -code : String -reservationDate : Calendar -price : float
+<<constructor>> ServiceReservationDto (id : Long, showId : long, email : String, creditCard : String, tickets : int, isValid : boolean, code : String, reservationDate : Calendar, price : float) : ServiceReservationDto

ServiceShowDto
-id : Long -name : String -description : String -startDate : Calendar -duration : long -limitDate : Calendar -tickets : Long -price : float -discountedPrice : float
+<<constructor>> ServiceShowDto (id : Long, name : String, description : String, startDate : Calendar, duration : long, limitDate : Calendar, tickets : Long, price : float, discountedPrice : float) : ServiceShowDto

ServiceShowAdminDto
-id: Long -name : String -description : String -startDate : Calendar -duration : long -limitDate : Calendar -maxTickets : long -tickets : Long -price : float -discountedPrice : float -commission : float
+<<constructor>> ServiceShowAdminDto(id : Long, name : String, description : String, startDate : Calendar, duration : long, limitDate : Calendar, maxTickets : long, tickets : Long, price : float, discountedPrice : float, commission : float) : ServiceShowAdminDto

3.2. REST

Selección de códigos de error:
Bad Request: Excepciones de peticiones mal formadas
Forbidden: Excepciones de lógica de negocio
Not Found: Excepciones con un identificador no encontrado

Casos de uso:

Crear evento

Método: POST
URL: remote/ws-app-service/shows
DTO (Entrada): ServiceShowAdminDto
DTO (Salida): ServiceShowDto
Respuestas:
400 Bad Request
201 Created

Actualizar evento

Método: PUT
URL: remote/ws-app-service/shows/{id}
DTO (Entrada): ServiceShowAdminDto
Respuestas:
400 Bad Request
403 Forbidden
404 Not Found
204 No Content

Buscar eventos por palabras clave

Método: GET
URL: remote/ws-app-service/shows
Parámetros:
keywords=<palabras separadas por espacios>
DTO (Salida): ServiceShowDto
Respuestas:
400 Bad Request
200 Ok

Obtener evento por id

Método: GET
URL: remote/ws-app-service/shows/{id}
DTO (Salida): ServiceShowDto
Respuestas:
400 Bad Request
404 Not Found
200 Ok

Obtener reservas de un usuario

Método: GET
URL: remote/ws-app-service/reservations
Parámetros:
email=<email de usuario>
DTO (Salida): Lista de ServiceReservationDto
Respuestas:
400 Bad Request
200 Ok

Marcar reserva

Método: PUT
URL: remote/ws-app-service/reservations/check
Parámetros:
code=<código de reserva>
creditCard=<número de tarjeta de crédito>
Respuestas:
400 Bad Request
403 Forbidden
404 Not Found
204 No Content

Comprar entradas

Método: POST
URL: remote/ws-app-service/reservations
Parámetros:
show=<código del evento>
email=<email del comprador>
creditCard=<número de tarjeta de crédito>
count=<número de entradas a comprar>
DTO (Salida): ServiceReservationDto
Respuestas:
400 Bad Request
403 Forbidden
404 Not Found
200 Ok

4. Aplicaciones cliente

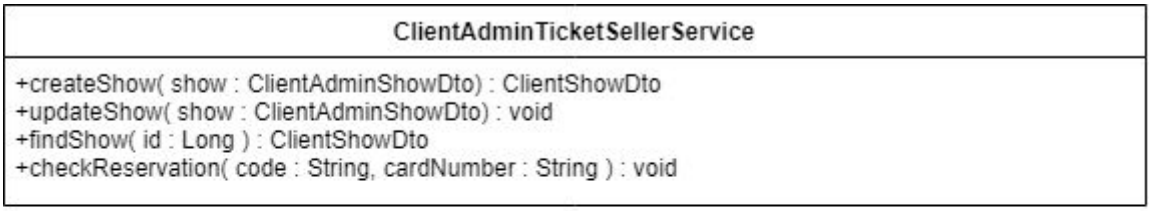
4.1. DTOs

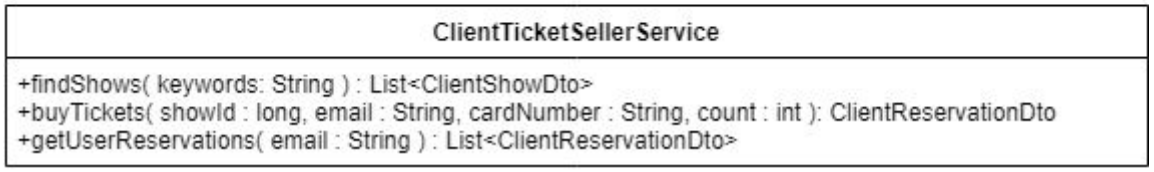
ClientAdminShowDto
-id : Long -name : String -description : String -startDate : Calendar -duration : long -limitDate : Calendar -maxTickets : long -tickets : long -price : float -discountedPrice : float -commission : float
+<<constructor>> ClientAdminShowDto (id : Long, name : String, description : String, startDate : Calendar, duration : long, limitDate : Calendar, maxTickets : long, tickets : long, price : float, discountedPrice : float, commission : float) : ClientAdminShowDto

ClientReservationDto
-id : Long -showId : Long -email : String -creditCard : String -tickets : int -isValid : boolean -code : String -reservationDate : Calendar -price : float
+<<constructor>> ClientReservationDto (id : Long, showId : long, email : String, creditCard : String, tickets : int, isValid : boolean, code : String, reservationDate : Calendar, price : float) : ClientReservationDto

ClientShowDto
-id : Long -name : String -description : String -startDate : Calendar -endDate : Calendar -limitDate : Calendar -tickets: long -price : float -discountedPrice : float
+<<constructor>> ClientShowDto (id : Long, name : String, description : String, startDate : Calendar, endDate : Calendar, limitDate : Calendar, tickets : long, price : float, discountedPrice : float) : ClientShowDto

4.2. Capa de acceso al servicio





* Un punto a comentar aquí es que dado que en los requisitos se especifica concretamente que se debe devolver un espectáculo sin comisión de venta ni el número máximo de entradas, esto implica que las peticiones de búsqueda de un espectáculo por un administrador no contienen todos los datos, pero sí que se crean espectáculos con todos los datos.

5. Trabajos tutelados

5.1 SOAP

5.2 Integración con una red social

6. Errores conocidos