Memoria

Sousa González, Carlos	carlos.sousa@udc.es
Hermida Mourelle, Pablo	pablo.hermnidam@udc.es
García Vázquez, Adrián	adrian.gvazquez@udc.es

Universidad de A Coruña
Facultad de Informática y Comunicación
Programación Avanzada

31 de mayo de 2020

Índice

1. Informe

En este informe vamos a reflejar la estructura de toda la práctica tanto el Backend como el Frontend, añadiendo diagramas para ayudar a su visualización. Adjuntamos una breve explicación de las decisiones que tomamos para afrontar la realización de la práctica.

1.1. Backend

1.1.1. Capa acceso a datos

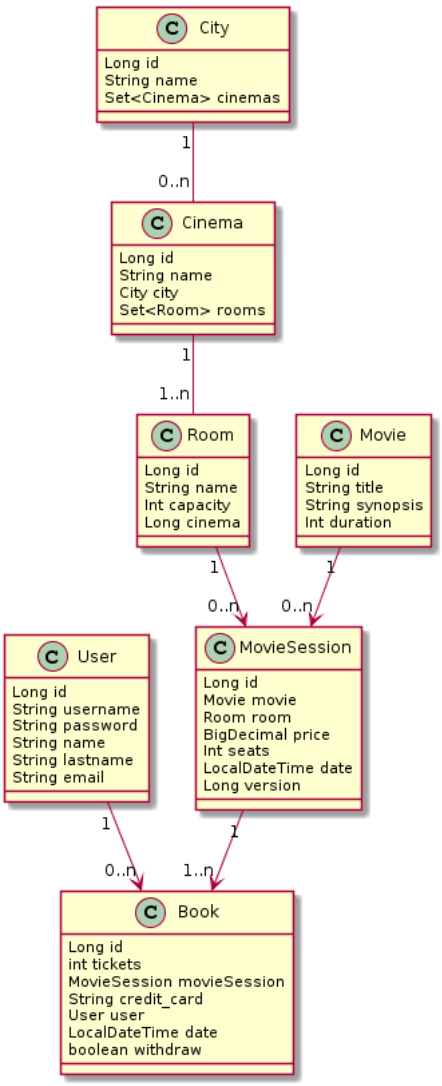

Figura 1: Diagrama de las Entidades

1.1.2. Capa lógica de negocio

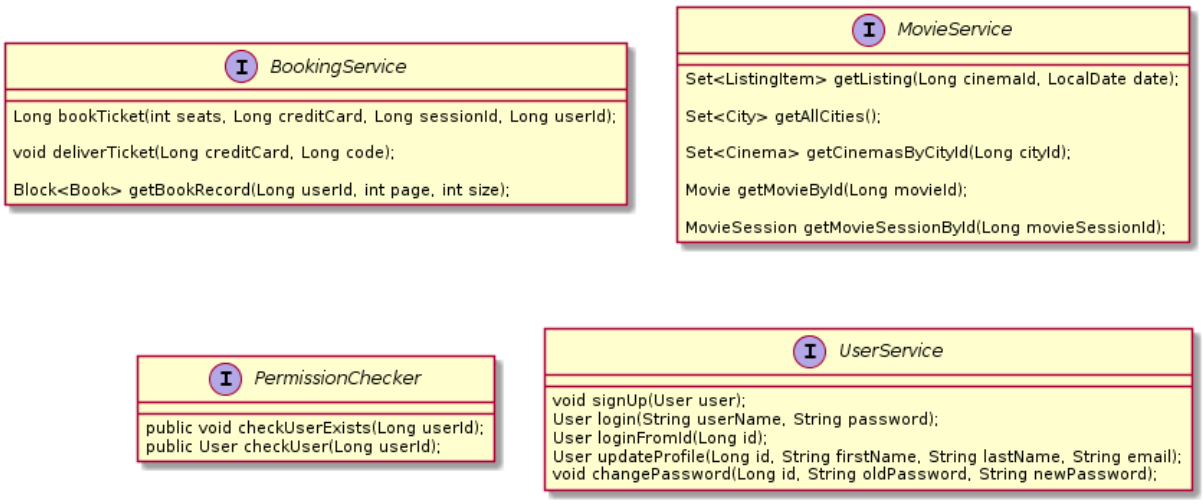

Figura 2: Diagrama de la Lógica de Negocio

Hemos dividido la lógica de negocio en 3 servicios diferentes (Booking, Movie y User) y un servicio auxiliar que sirve de apoyo para el UserService. En el caso de BookingService, agrupamos los casos de uso relacionados con la venta de entradas y la entrega de tickets por parte del taquillero. En MovieService englobamos todos los accesos necesarios para obtener los datos relacionados con las películas y los cines. Finalmente en UserService tenemos los casos de uso relacionados con el registro de los usuarios.

En MovieSession hemos creado la propiedad seats inicializándola con el valor de la capacidad de la sala. En este caso, lo usaremos para ir contabilizando cuantas entradas nos quedan disponibles en cada sesión.

Para el caso de reservar una entrada, hemos añadido la técnica Optimistic Locking, ya que dos personas reservando entradas a la vez podrían conseguir más entradas que capacidad tiene una sala. Para ello hemos añadido el atributo versión a la entidad MovieSession.

1.1.3. Capa servicios REST

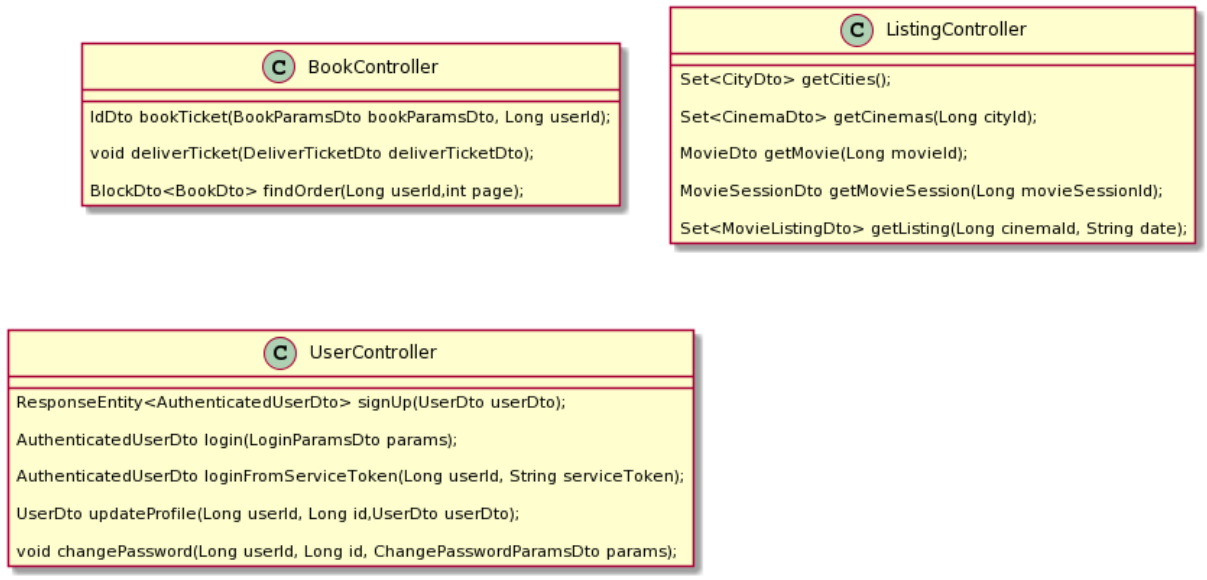

Figura 3: Diagrama de los Controladores

Se ha creado un controlador por cada servicio al que se va a acceder desde el frontend (Booking, Movie y User). No es necesario crear controlador para el servicio auxiliar debido a que no se va a acceder desde el frontend.

1.2. Frontend

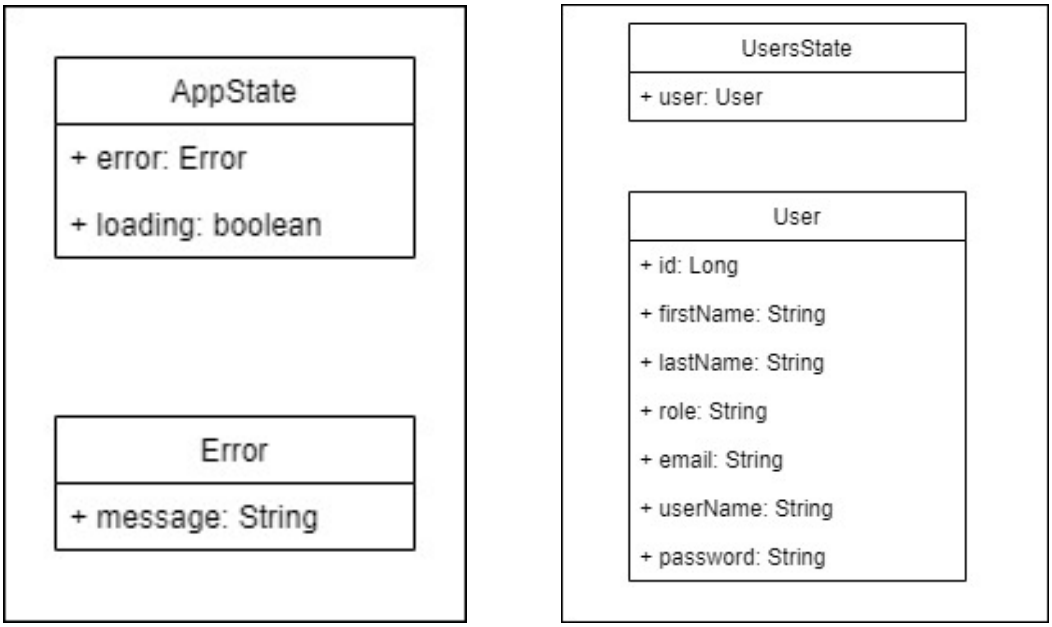

Figura 4: Diagramas de estado módulo App y Users

Para este módulo destacamos la creación del estado Ticket, que para el Backend no se corresponde con ninguna entidad definida.

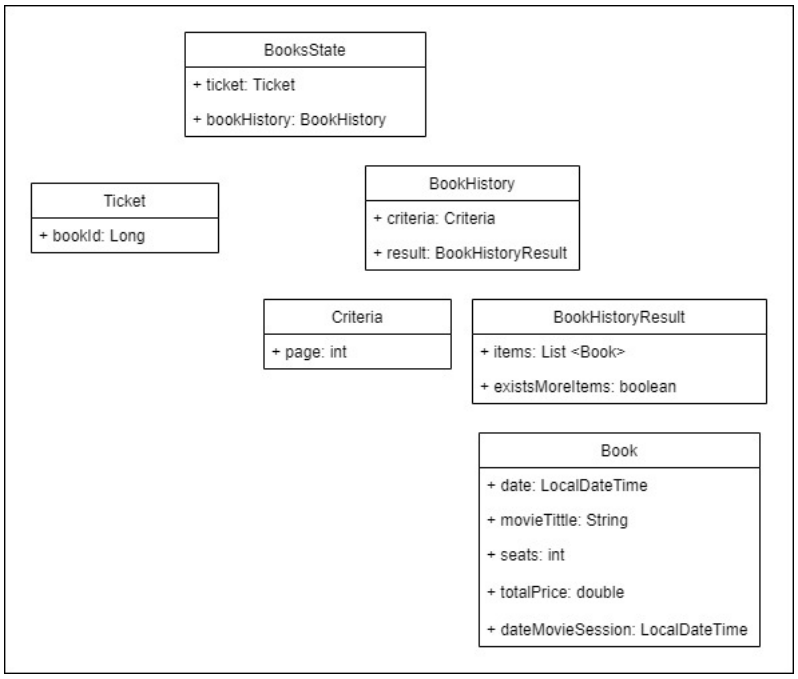

Figura 5: Diagrama de estado módulo Books

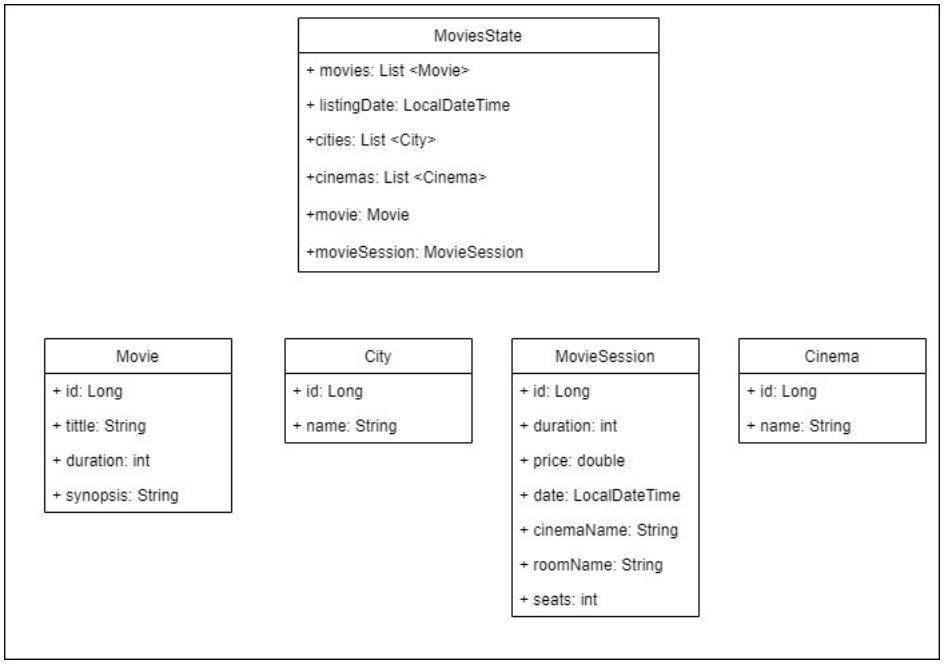

Figura 6: Diagrama de estado módulo Movies

1.3. Trabajos tutelados

- BatchSize

  Para la implementación de este apartado se ha añadido la anotación **@BatchSize(size=10)** en las siguientes entidades: *Cinema, Movie y MovieSession* con el fin de prevenir el problema de los *n+1 SELECTS* al buscar los cines de una ciudad, las peliculas de un cine y las sesiones de cada película. Esta optimización nos sirve para inicializar un grupo de proxies en la misma operación SELECT, en lugar de solo una, reduciendo el número de peticiones que se hacen a la base de datos. Se ha usado el tamaño 10, el cual no es muy alto y coincide con la paginación usada en el trabajo.

  Para la búsqueda de las Sesiones de un Cine de la misma Película se necesita aplicar **BatchSize** a estras tres entidades, debido al problema mencionado anteriormente. Gracias a esta anotación nos permite simplificar todas estas consultas para evitar un gran número de SELECTS.

- Test Unitarios Frontend

  Al realizar los test, tuvimos que modificar el componente BuyForm, y retirar la comprobación de si el usuario estaba logeado, esta comprobación ya la realiza el componente que carga el BuyForm(MovieSessionView).

  Y a su vez para el test de *actions* hemos exportado la action de **buyTicketCompleted** para poder testearla.

1.4. Problemas conocidos

A la hora de mostrar alertas, cuando se ejecuta el cierre de ellas en algunas vistas quedan ocultas para siempre a no ser que se vuelva a cargar el componente de nuevo. Para arreglar esto, decidimos modificar el componente DeliverTicket estableciendo un timeout a las alertas, ya que es donde vimos este problema puntual y así no modificamos el componente Errors.

No es un problema como tal, pero queríamos recalcar el uso y la modificación del componente **Success** del módulo Common para mostrar mensajes informativos sobre entrega de Tickets y selección correcta de cine favorito.