Notas y consideraciones de la práctica

Consideraciones técnicas:

1. Se ha hecho uso de una librería propia para realizar consultas genéricas con Hibernate, desde el controller se llama al DAO que ya trata con esta librería. (excepto un find where).
2. Ambas clases son abstractas con métodos estáticos.
3. Se ha usado una librería de María del Mar para los inputs
4. Hay 3 tipos de excepciones, las generadas por la base de datos: DBException, generadas por errores del usuario: UserException y por último excepciones críticas de la aplicación: ApplicationException, éstas última es la única que cierra la aplicación por completo. Al no poder detectar un error de conexión entre Hibernate y la DB ya que Hibernate no devolvía ninguna excepción (a diferencia de la SQLException de JDBC) no se ha podido comprobar con exactitud éste error.

Consideraciones funcionales:

1. Se ha usado la matrícula para acceder a los Usuarios ya sea para edición como borrado, mientras que los Expedientes al no disponer de una clave de negocio dichas funcionalidades han sido implementadas con la id de la base de datos (clave primaria)
2. Por énde, y dado que en la base de datos no se considera única, se comprueba que la matrícula de un Usuarios sea única. (Hibernate ya comprueba la id de Expedientes)

Consideraciones relacionadas con el diseño:

1. Se han implementado las clases MenuOption y MenuService para poder trabajar con objetos respecto al menú de selección del usuario. Dicho menú se genera en MenuService y luego se comprueba si el usuario tiene el tipo indicado para dicha Action. En caso de querer implementar una nueva opción sólo se debería crear una MenuOption en MenuService e implementar su funcionalidad en el controlador.
2. Al modificar el tipo del propio usuario logueado se desloguea automáticamente ya que debe volver a cargar las opciones del menú. (se sale al momento y no deja modificar más ya que si pasa de ser ADMIN a AUX éste ya no tendría permisos para modificar)
3. Se crean y cierran sesiones de Hibernate en cada consulta
4. En las consultas con Hibernate se guarda el resultado en una variable en vez de hacer return directamente porque se necesita cerrar la sesión antes e Hibernate no permite realizar consultas sin sesión
5. Se ha configurado Hibernate en fetching/loading al tipo Eager, de esta forma al obtener un objeto carga consigo sus colecciones ya que no se ha considerado que el tamaño de dichas colecciones afecte al rendimiento en una aplicación pequeña
6. Se ha decidido poner el enum UserPerfil fuera de Usuarios pese a que pertenece a dicha clase para modificar en lo menor posible los POJOs de Hibernate
7. Dicho enum permite trabajar todo el rato con valores UserPerfil.ADMIN en vez de ints, si se quiere implementar un nuevo tipo de usuario sólo se debe añadir a dicho enum y la aplicación se controlaría sola

Alumno: Àlex Cuevas Cervilla

Práctica: VetStucom (Acceso a datos)

Fecha: 12/02/2019