20/01/26

ScrumDesk:

- En la historia de usuario “Ver eventos en listado” no has hecho los cambio que te indiqué:
    - En el paso 2 deberías hacer referencia a que el listado de eventos se muestra en el panel lateral (¿es así, no?).
    - Los pasos 2, 3 y 4 en realidad tendrían que ser uno solo.

Modelos UML

- No se modela explícitamente la base de datos en el diagrama de componentes.
- Los Repository yo los llamaría simplemente <>Repository, no <>RepositorySQL porque en realidad el mismo código te va a valer para diferentes bases de datos.
- Te he separado la arquitectura en dos diagramas: el de componentes conectados por interfaces, y otro con el detalle de las interfaces.
- En las interfaces de la capa Service, por el momento, pon solo los métodos que te hacen falta para las historias de usuario que vas a implementar. En este caso sería solo registraUsuario y listarEventos (que por cierto, yo lo llamaría solo eventos, ya que este método no lista nada, solo retorna los eventos). En el caso de las interfaces Repository sí podrías poner todas las interfaces, porque en realidad tienes todos los métodos.
- Repasa porque incluso solo en estos métodos que te digo, faltan parámetros.
- Creo que ComentariosController, ComentariosService y ComentariosRepository sobran. Por un lado, en este Sprint sobran porque no vamos a implementar funcionalidad de comentarios, y por otro, porque a los comentarios nunca vas a acceder directamente, sino a través de su correspondiente Evento, así que con el EventoService valdría, lo que sí podrías añadir es un método anhadeComentarioAEvento, por ejemplo.

15/01/26

ScrumDesk:

- En la historia de usuario “Ver eventos en listado” no has hecho los cambio que te indiqué:
    - En el paso 2 deberías hacer referencia a que se muestra en el panel lateral. Y en ese mismo incluir lo que pones en el caso 4 (por cada evento se muestra…)
    - El paso 3 sobra.

Modelos UML

- Te he cambiado un poco la estructura del modelo, separando el dominio de la arquitectura.
- También he cambiado el nombre del propio archivo.
- Modelo de dominio:
    - Los id mejor que sean long.
    - Las relaciones entre Evento e ImagenEvento y Evento y Comentario ¿son bidireccionales? Según el diagrama lo son, pero me extraña (tendrías que poner la flechita).

    - En general repasa la navegabilidad de acuerdo a como la implementes en realidad.
    - El atributo contraseña debería guardarse codificado.
- Arquitectura
    - El diagrama que usas para la arquitectura no es el adecuado, tiene que ser un diagrama de componentes, donde cada Repository, Service y Controller sean un componente. Te he iniciado el diagrama con algún componente, tendrías que rellenar el resto.
    - Al igual que haces en los Repository, los servicios también tienen que ofrecer su funcionalidad a través de una interfaz (aunque luego en código no lo hicieses así). Para hacer esto, las operaciones que tienes ya creadas en las clases Service las puedes arrastrar directamente (en la vista de árbol del modelo) a las nuevas interfaces que vayas creando.

12/12/25

ScrumDesk:

- Registro de usuario
    - ¿Por qué hay que pulsar iniciar sesión primero? ¿De inicio no hay para registro?
    - Entre el paso 2 y el 3 supongo que falta uno del estilo “El sistema muestra un formulario con los campos para el registro: nombre, etc.”
    - Falta el caso de fallo en acceso a base de datos.
- Inicio de sesión
    - En la historia de Registro, cuando era válido, ponía “La aplicación muestra la pantalla principal con el nombre del usuario arriba a la derecha.”, aquí pone “La aplicación muestra el nombre del usuario arriba a la derecha.” Deberían ser iguales, ¿no?
    - Falta el caso de fallo en acceso a base de datos.
- Ver eventos en listado
    - En el paso 2 deberías hacer referencia a que se muestra en el panel lateral. Y en ese mismo incluir lo que pones en el caso 4 (por cada evento se muestra…)
    - El paso 3 sobra.

Modelos UML:

- En el modelo de dominio te he puesto comentarios en una nota en el propio diagrama.
- Haz un solo modelo con todo, no crees uno para el dominio y otro para la arquitectura, porque en el segundo tienes que usar el primero.
- El modelo de arquitectura no lo entiendo, hablas de Presenter y Repository, cuando aquí vas a usar React, no Android. Además, esa arquitectura debería ser la del backend.
- En definitiva, el modelo que vas a hacer de momento con UML es el del back, si quieres podrías crear otro para el front, y ese sí sería en un modelo StarUML diferente.