Estrategias de testing

Reporte

1. **Testing de creación:** Probar que un nuevo reporte se cree exitosamente en la base de datos con todos los campos necesarios: reporterUserId, reportedUserId, reportReason, etc. Verificar que el ID se genere correctamente y que el estado inicial sea "Pendiente" o el valor predefinido.
2. **Testing de lectura:** Comprobar que se puede recuperar un reporte existente por su reportId. Esto incluye verificar que todos los datos asociados se devuelvan correctamente, incluyendo el reportReason, reportDate, y cualquier comentario adicional.
3. **Testing de actualización:** Validar que se puedan actualizar los detalles de un reporte existente, como cambiar el reportStatus de "Pendiente" a "Revisado". Probar que los cambios se guarden y se reflejen correctamente en el reporte.
4. **Testing de eliminación:** Asegurarse de que un reporte pueda ser eliminado sin problemas y que, tras la eliminación, el reporte no esté accesible. También se debe verificar que las referencias o contadores en otras entidades se actualicen adecuadamente.
5. **Testing de validación:** Revisar que los campos del reporte cumplan con las reglas de validación. Por ejemplo, reportDate debe ser una fecha válida y no futura, reportReason debe tener un valor permitido (como "spam" o "lenguaje ofensivo"), y los IDs de usuario deben existir en la base de datos.

Usuario

1. **Testing de creación:** Asegurarse de que un usuario se cree correctamente en la base de datos con todos los campos necesarios: username, email, y registrationDate. Validar que el estado inicial de la cuenta sea "Activa" y que userId se genere correctamente.
2. **Testing de lectura:** Verificar que se pueda acceder a un usuario existente utilizando su userId y que todos los datos, como username y accountStatus, se devuelvan correctamente.

3. **Testing de actualización:** Comprobar que se puedan actualizar datos de un usuario, como accountStatus de "Activa" a "Suspendida" o cambios en el email. Verificar que los datos modificados se guarden adecuadamente.
4. **Testing de eliminación:** Probar que un usuario pueda ser eliminado y que, tras la eliminación, no se pueda acceder a su información. También revisar que las referencias de sus publicaciones o reportes se manejen correctamente.
5. **Testing de validación:** Asegurar que los campos del usuario cumplan con las reglas de validación. Por ejemplo, email debe seguir el formato de correo, username debe cumplir con la longitud mínima y máxima, y accountStatus solo debe tener valores permitidos como "Activa" o "Suspendida".

Publicación

1. **Testing de creación:** Probar que una nueva publicación se cree con éxito en la base de datos con todos los campos obligatorios: authorId, content, y createdAt. Verificar que se generen correctamente el postId y que el estado inicial de visibilidad sea "Pública" o el valor predeterminado.
2. **Testing de lectura:** Validar que se pueda recuperar una publicación existente por su postId y que toda la información se devuelva correctamente, incluyendo content, visibility, y reportsCount.
3. **Testing de actualización:** Probar que se puedan actualizar los detalles de una publicación, como cambiar el visibility de "Pública" a "Privada". Verificar que los cambios se guarden y se reflejen adecuadamente.
4. **Testing de eliminación:** Asegurarse de que una publicación pueda ser eliminada y que no sea accesible tras la eliminación. Verificar que, al eliminar la publicación, cualquier referencia o contador en otras entidades se actualice adecuadamente.
5. **Testing de validación:** Revisar que los campos de la publicación cumplan con las reglas de validación. Por ejemplo, content debe tener un tamaño adecuado y visibility debe tener un valor permitido como "Pública" o "Privada".

Testing de Integración

1. **Asociación correcta de reportes:** Validar que un reporte esté correctamente asociado a un usuario reportante y a la publicación (si corresponde) o usuario reportado. Asegurarse de que el sistema permita un seguimiento adecuado de cada entidad.
2. **Condiciones de cuenta del usuario:** Verificar que solo usuarios con cuentas activas puedan crear publicaciones y enviar reportes. Comprobar que usuarios suspendidos o inactivos no tengan permisos para realizar estas acciones.
3. **Validación de la publicación:** Asegurarse de que una publicación tenga asignado un autor válido (authorId) y que el campo content cumpla con los requisitos antes de ser guardado o editado en la base de datos.

Testing de Seguridad

1. **Autorización en reportes:** Comprobar que solo los usuarios autorizados puedan crear, leer, actualizar y eliminar reportes. Asegurarse de que los usuarios no autorizados reciban respuestas de error adecuadas.
2. **Autorización en usuarios:** Asegurarse de que solo usuarios autorizados puedan crear, leer, actualizar y eliminar cuentas de usuario. Verificar que se implementen controles de acceso adecuados para proteger la privacidad de los datos de usuario.
3. **Autorización en publicaciones:** Verificar que solo usuarios autorizados puedan gestionar publicaciones. Validar que los usuarios sin permisos adecuados no puedan acceder ni modificar publicaciones de otros usuarios.

Herramientas de Testing

- **Mockito:** Framework de simulación que permite crear objetos simulados y probar la interacción entre componentes en Spring Boot. Es útil para pruebas unitarias en las que se requiere simular dependencias.
- **Spring Boot Test**: Proporciona una configuración de testing integral para Spring Boot, incluyendo anotaciones específicas *(@SpringBootTest, @WebMvcTest, etc*.) que facilitan las pruebas de integración y la configuración del contexto de la aplicación.

Proceso de Testing

1. **Planificación:** Definir el alcance y los objetivos de las pruebas de acuerdo con los requisitos del sistema de reportes, usuarios y publicaciones.
2. **Preparación:** Configurar el entorno de testing y preparar los datos necesarios, como usuarios de prueba y publicaciones.
3. **Ejecución:** Ejecutar las pruebas de acuerdo con los casos definidos, verificando que el sistema cumpla con los requisitos de funcionalidad, integración y seguridad.
4. **Evaluación:** Revisar los resultados y registrar defectos si se encuentran problemas durante la prueba.
5. **Corrección:** Realizar correcciones en el sistema y ejecutar pruebas de regresión para asegurar que los defectos se hayan solucionado sin introducir otros problemas.