Extensiones

Autor: Daniel Morales

Proyecto: Juan Jose Garcia - Biblioteca Pingu

Extensiones realizadas:

1. **Creación de la clase RegistroUsuarios**:
    - Clase para manejar el registro de usuarios, incluyendo la creación de superusuarios, administradores, asistentes y lectores.
2. **Creación de métodos de registro**:
    - `registrarSuperUsuario()`: Método para registrar un superusuario en el sistema.
    - `registrarAdministrador()`: Método para registrar un administrador.
    - `crearAsistente()`: Método para crear un asistente.
    - `crearLector()`: Método para crear un lector.
3. **Actualización de información del usuario**:
    - `actualizarInformacionUsuario()`: Método para permitir a los usuarios actualizar su información.
    - `obtenerUsuarioPorCorreo()`: Método para obtener un usuario por su correo electrónico.
4. **Gestión de préstamos de prueba**:
    - Clase `PrestamosPrueba`: Clase para gestionar y simular préstamos de prueba.
    - Creación de menú de simulación de préstamos de prueba.
5. **Modificaciones a la tabla de usuarios**:

- Adición de campos adicionales en la tabla de usuarios: `telefono` y `direccion`.
- Adaptación de los métodos CRUD de usuarios para trabajar con los nuevos campos.
- Actualización de la clase Usuario para incluir los nuevos campos.

6. **Extensiones de la clase Producto**:
   - Expansión del atributo `tipo` para representar diferentes tipos de productos como video grabaciones, canciones y ensayos tipo tesis.
7. **Gestión de IO (input & output)**:
   - Creación de la carpeta exporter_importer para manejar la importación y exportación de inventario en formatos XML, JSON y CSV.
   - Clases especializadas para la importación y exportación de archivos según su tipo (CSV, XML, JSON).
8. **Creación de clases para manejo de inventario**:
   - Clases especializadas en la exportación de archivos de inventario (CSV, XML, JSON).
   - Clases especializadas en la importación de archivos de inventario (CSV, XML, JSON).
9. **Método `getPreparedStatement` en MySQLoperations**:
   - Creación de un método para preparar declaraciones SQL con parámetros (`getPreparedStatement`), facilitando la interacción con la base de datos.

Trigger

Creé una tabla llamada registros_creados que almacena información sobre los usuarios recién creados. Esta tabla contiene un identificador único, quién los creó (por defecto 'SUPERSUARIO'), el correo del nuevo usuario, la fecha y hora de creación, y la acción 'Usuario creado'. También diseñé un trigger registros_creados que se dispara después de insertar un nuevo registro en la tabla usuario, verificando si el rol del nuevo usuario no es 'LECTOR'. Si el rol es diferente, el trigger inserta un registro en la tabla registros_creados con la

información de quién creó al nuevo usuario, el correo del nuevo usuario, y la acción 'Usuario creado'.

Conexión con bases de datos relacionales y NoSQL:

Se creó una clase de conexión para MongoDB, permitiendo su funcionamiento simultáneo con la base de datos relacional MySQL. También se desarrollaron clases de migración para cada tabla de la base de datos relacional, lo que facilita la sincronización de la información entre las dos bases de datos y previene la migración de elementos duplicados.

Conclusión:

La documentación muestra una serie de extensiones realizadas en el sistema, que incluyen mejoras en el registro de usuarios, actualización de información, gestión de préstamos, y manejo de entrada y salida de datos. También se destaca la implementación de triggers y la integración de bases de datos relacionales y NoSQL para optimizar el flujo de información. Estas mejoras fortalecen la funcionalidad y la eficiencia del sistema en su conjunto.

Video funcionalidades

https://www.youtube.com/watch?v=dCE1mlS5_y0