

Informe Técnico - Arquitectura y Estándares

Sistema de Control de Acceso (QR + SQLite)

Versión: v1.0 | Proyecto: Grupo 1 | Fecha: 2026-01-27

1. Propósito del sistema

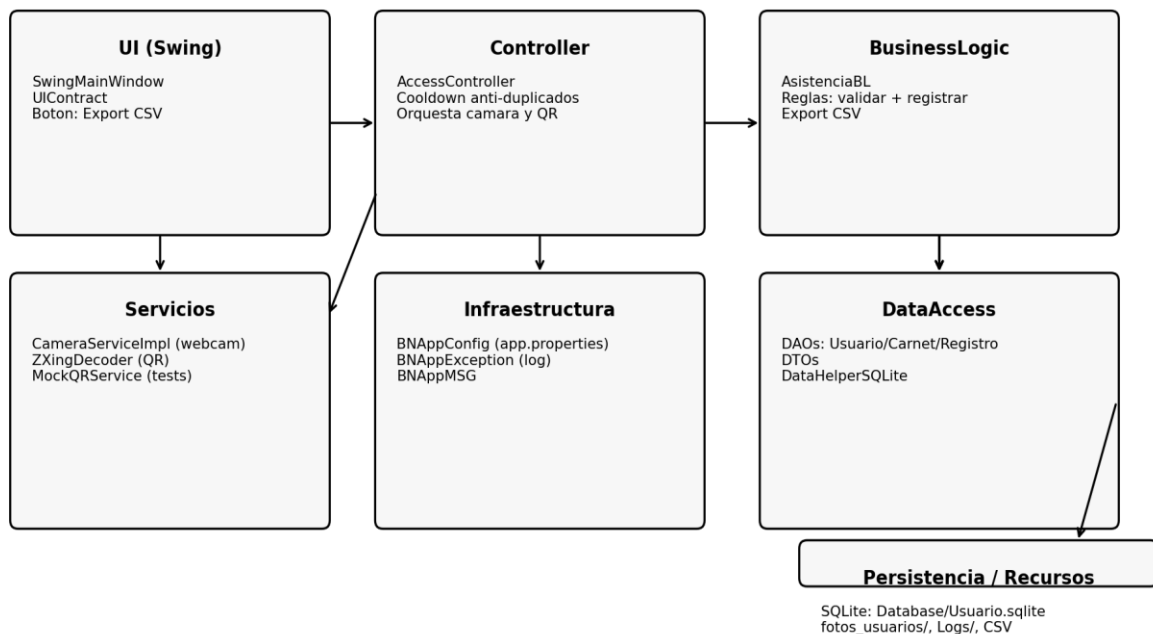
Aplicación de escritorio en Java (Swing) para registrar accesos mediante lectura de códigos QR con cámara web. El sistema valida el QR contra una base de datos SQLite (Usuario + Carnet) y registra el acceso en la tabla Registro. Adicionalmente permite exportar un reporte a CSV.

2. Arquitectura (MVC + Capas)

Se aplica una separación por capas: UI (Swing), Controlador, Lógica de Negocio (BL), Acceso a Datos (DAO/DTO) e Infraestructura (configuración y logs).

2.1 Diagrama de arquitectura

Diagrama basado en la estructura real del código (paquetes en /src).



2.2 Flujo principal de acceso

- App.main inicia SwingMainWindow y crea AccessController.
- AccessController inicia CameraService y ejecuta un loop (ScheduledExecutor) ~30 FPS.
- Por cada frase: se actualiza la UI (SwingUtilities) y se intenta decodificar QR (ZXingDecoder).
- Si hay QR y pasa cooldown: AsistenciaBL.registrarAcceso(qr).
- AsistenciaBL valida Carnet y Usuario (Estado = A) y registra en Registro.
- UI muestra acceso garantizado/denegado y reproduce beep en exito.
- Botón Exportar CSV: AccessController -> AsistenciaBL -> AsistenciaCsvExporter.

2.3 Mapa de paquetes y responsabilidades

- UI: SwingMainWindow (vista principal) + UIContract (contrato).
- Controller: AccessController (orquesta cámara, QR y BL; controla cooldown y actualiza la vista).

- Camera: CameraService/CameraServiceImpl (webcam-capture).
- QR: ZXingDecoder (ZXing) y servicios de QR para pruebas (MockQRService).
- BusinessLogic.Entities: AsistenciaBL (caso de uso registrar acceso + exportar CSV).
- DataAccess: DAOs (UsuarioDAO, CarnetDAO, RegistroDAO, CatalogoDAO) y DTOs.
- Infrastructure: BNAppConfig (properties), BNAppException (logging), BNAppMSG (mensajes).
- Resources: Database/Usuario.sqlite, fotos_usuarios/, salida CSV, Logs/.

3. Estándares de desarrollo observados y recomendados

3.1 Estructura y capas

- UI solo conoce UIContract y al Controller (no ejecuta SQL ni reglas de validacion).
- Controller concentra el loop de camara + lectura QR y delega validacion a BL.
- BL encapsula reglas (validacion de estados, registro de asistencia, export).
- DAO/DTO encapsulan SQL y objetos de transferencia; no contienen logica de UI.

3.2 Convenciones de codigo

- Clases: PascalCase (UsuarioDAO, AsistenciaBL).
- Metodos: camelCase (readById, registrarAcceso).
- Constantes: MAYUSCULAS con guion bajo (COOLDOWN_MS).
- SQL parametrizado: PreparedStatement para entradas (evita inyeccion SQL).

3.3 Manejo de errores y logging

- Errores tecnicos de infraestructura/BD se encapsulan en BNAppException, que escribe log en archivo.

- Errores de validacion funcional (QR no registrado, usuario inactivo) se devuelven como mensajes (sin stacktrace).
- Recomendacion: separar ExcepcionTecnica vs ExcepcionValidacion para no registrar como error casos esperados.

3.4 Concurrencia y UI (Swing)

- El loop de lectura corre en ScheduledExecutor (hilo secundario).
- Toda actualizacion visual se ejecuta via SwingUtilities.invokeLater (hilo EDT).
- Cooldown para evitar registros duplicados por lecturas consecutivas del mismo QR.

3.5 Configuracion y rutas

- Parametros centralizados en src/app.properties: path.database y path.log.
- Recomendacion Maven: mover app.properties a recursos (src/main/resources) y leerlo por ClassLoader.
- Recomendacion multiplataforma: usar '/' en path.log (ej. src/Logs/asistencia_error.log).
- Recomendacion de recursos: la BD debe guardar solo el nombre de archivo (maria.jpg) y la UI resolver con fotos_usuarios/.

3.6 Dependencias y build

- Gestión por Maven (pom.xml) para: webcam-capture, zxing (core + javase), sqlite-jdbc, slf4j-simple.
- Recomendación: si se entrega un JAR ejecutable, alinear pom.xml: mainClass debe apuntar a 'App' (o crear AppBootstrap).
- Recomendación: evitar duplicar librerías en /lib si ya se usa Maven.

4. Estándares de base de datos (SQLite)

La BD usa diccionarios (CatalogoTipo/Catalogo) para reutilizar valores (rol, sexo, estado civil, raza) y entidades principales Usuario, Carnet y Registro.

4.1 Reglas de integridad y auditoria

- PRAGMA foreign_keys = ON para forzar integridad referencial.
- Campo Estado (A/X) para borrado logico (soft delete).
- Campos FechaCreacion y FechaModificacion para auditoria.
- Carnet.CodigoQR es UNIQUE (no se repite).
- Indice UNIQUE ux_carnet_usuario para garantizar 1 carnet por usuario.

4.2 Esquema (tablas y columnas)

Tabla: Carnet

Columna	Tipo	NOT NULL	PK	Default
IdCarnet	INTEGER	NO	SI	
IdUsuario	INTEGER	SI	NO	
CodigoQR	VARCHAR(75)	SI	NO	
Estado	VARCHAR(1)	SI	NO	'A'
FechaCreacion	DATETIME	NO	NO	datetime('now','localtime')
FechaModificacion	DATETIME	NO	NO	

Tabla: Catalogo

Columna	Tipo	NOT NULL	PK	Default
IdCatalogo	INTEGER	NO	SI	
IdCatalogoTipo	INTEGER	SI	NO	

Nombre	VARCHAR(50)	SI	NO	
Descripcion	VARCHAR(90)	NO	NO	
Estado	VARCHAR(1)	SI	NO	'A'
FechaCreacion	DATETIME	NO	NO	datetime('now','localtime')
FechaModificacion	DATETIME	NO	NO	

Tabla: CatalogoTipo

Columna	Tipo	NOT NULL	PK	Default
IdCatalogoTipo	INTEGER	NO	SI	
Nombre	VARCHAR(20)	SI	NO	
Descripcion	VARCHAR(90)	NO	NO	
Estado	VARCHAR(1)	SI	NO	'A'
FechaCreacion	DATETIME	NO	NO	datetime('now','localtime')
FechaModificacion	DATETIME	NO	NO	

Tabla: Registro

Columna	Tipo	NOT NULL	PK	Default
IdRegistro	INTEGER	NO	SI	
IdUsuario	INTEGER	SI	NO	
IdCarnet	INTEGER	SI	NO	
FechaEntrada	DATETIME	NO	NO	datetime('now','localtime')
Estado	VARCHAR(1)	SI	NO	'A'

Tabla: Usuario

Columna	Tipo	NOT NULL	PK	Default
IdUsuario	INTEGER	NO	SI	
IdCatalogoTipoUsuario	INTEGER	SI	NO	
IdCatalogoSexo	INTEGER	SI	NO	
IdCatalogoEstadoCivil	INTEGER	SI	NO	
IdCatalogoRaza	INTEGER	SI	NO	
PrimerNombre	VARCHAR(15)	SI	NO	
SegundoNombre	VARCHAR(15)	NO	NO	
PrimerApellido	VARCHAR(12)	SI	NO	
SegundoApellido	VARCHAR(12)	SI	NO	
Cedula	VARCHAR(10)	SI	NO	
Foto	TEXT	SI	NO	'sin_foto.jpg'
Estado	VARCHAR(1)	SI	NO	'A'
FechaCreacion	DATETIME	NO	NO	datetime('now','localtime')
FechaModificacion	DATETIME	NO	NO	

4.3 Datos semilla y consistencia de recursos

- CatalogoTipo/Catalogo definen valores de rol, sexo, estado civil y raza.
- Consistencia recomendada: en Usuario.Foto guardar nombres existentes en fotos_usuarios/ (ej. 'maria.jpg').
- Para pruebas: un Carnet por usuario;CodigoQR debe coincidir exactamente con el texto del QR real.

5. Checklist de entrega (para integracion)

- 1) Ejecutar Script/DDL.sql y Script/DML.sql (o validar que Database/Usuario.sqlite ya tenga los datos).
- 2) Confirmar que fotos_usuarios/ contenga las fotos usadas en Usuario.Foto.
- 3) Abrir la app desde App.java (App.main) y verificar que la camara se abra.
- 4) Probar lectura QR con un QR que contenga exactamente unCodigoQR existente en Carnet.CodigoQR.
- 5) Verificar que se inserte una fila en Registro y que Exportar CSV genere el archivo solicitado.