

ENTREGA FINAL – Proyecto 3 – Interfaz gráfica, QR e impresión de tiquetes

Proyecto 3 – Entrega final: Interfaz gráfica, impresión de tiquetes y generación de QR

Facultad de Ingeniería, Universidad de los Andes

ISIS 1226: Diseño y programación orientada a objetos

Hector Florez

7 de Diciembre 2025

Sistema: Boletamaster - GUI

Integrantes:

- Tomás Fernando Albarracín Malaver - 202421942
 - Juan Esteban García Bonillas - 202420481
 - Juan Esteban Angel Perdigon - 202420145
-

Introducción

En esta entrega final del proyecto Boletamaster, se migró completamente la interacción del sistema desde consolas independientes hacia una interfaz gráfica unificada, implementada con Java Swing bajo una arquitectura modular que mantiene el principio de *bajo acoplamiento y alta cohesión*.

El sistema ahora permite a los usuarios iniciar sesión, gestionar eventos, comprar tiquetes, visualizar tiquetes adquiridos, imprimirlos y generar códigos QR, todo desde un entorno visual.

Además, se integró la librería ZXing para la generación de códigos QR y se adaptaron las clases de dominio para soportar procesos como registro de impresión, asociación de eventos/localidades, y visualización estructurada en tablas.

El proyecto conserva el modelo de persistencia mediante JSON para asegurar que usuarios, eventos y tiquetes sean cargados y guardados entre ejecuciones.

1. Estado de la Aplicación

1.1 Interfaz gráfica (GUI)

La aplicación ahora se ejecuta desde una ventana principal (**MainApp**), que redirige al usuario según su rol:

- **AdminFrame**
- **OrganizadorFrame**
- **ClienteFrame**

Cada una implementa menús, paneles y flujos totalmente controlados por componentes Swing como:

- JTable para listar datos
- JOptionPane para entradas y confirmaciones
- JFileChooser / paths locales para gestionar imágenes de QR
- CardLayout para cambiar vistas internas

1.2 Funcionalidades nuevas de la GUI

- Inicio de sesión visual.
- Compra de tiquetes mediante selector de eventos/localidades.
- Panel de “Mis tiquetes” con tabla dinámica.
- Impresión de tiquetes (registro de fecha/hora).
- **Generación automática de códigos QR** asociados al tiquete.
- Actualización visual del saldo virtual.

- Manejo de errores controlados en la UI.

1.3 Ajustes al dominio

Se extendió la clase Tiquete para incluir:

- evento
- localidad
- comprador
- fechaImpresion

These campos permiten que el sistema GUI muestre información completa y genere QR con datos integrados.

2. Objetivos

Objetivo general

Implementar una interfaz gráfica funcional para Boletamaster que permita comprar, visualizar, imprimir y generar códigos QR de tiquetes, conservando la estructura modular del sistema y la persistencia mediante JSON.

Objetivos específicos

- Desarrollar una interfaz gráfica clara para cada rol del sistema.
- Permitir compra y manejo de tiquetes desde la GUI.
- Implementar generación de QR usando ZXing.
- Registrar fecha/hora de impresión de tiquetes.
- Mostrar los tiquetes del cliente en una tabla interactiva.
- Mantener compatibilidad con la persistencia previa del proyecto.
- Garantizar validez de datos y reglas de dominio desde la interfaz.

3. Reglas de Dominio (Actualizadas)

1. Todo usuario debe autenticarse mediante login y contraseña antes de usar la aplicación.
2. El saldo virtual es el único medio de pago permitido.
3. El organizador solo puede crear venues y eventos.
4. Un venue debe estar aprobado para poder alojar eventos.
5. Un evento debe tener fecha, hora, tipo y localidad(es) válidas.
6. La compra de tiquetes solo es posible para clientes con saldo suficiente.
7. Un tiquete solo puede imprimirse una vez.
8. Un tiquete solo genera QR si ya está impreso.
9. La tabla “Mis Tiquetes” refleja siempre el estado actual del cliente.
10. La persistencia debe conservar usuarios, eventos, y tiquetes entre ejecuciones.
11. El sistema no debe permitir objetos inconsistentes ni nulos en la interfaz.
12. Cada tiquete tiene un QR único derivado de su ID.

4. Diagramas Actualizados

Los diagramas se encuentran en formato SVG en el repositorio en GitHub.

5. Implementación

5.1 Arquitectura en capas

- **Capa de dominio:** Maneja la lógica central (Tiquete, Usuario, Evento, Localidad).

- **Capa de persistencia:** JsonStore serializa/deserializa toda la información del sistema.
- **Capa de presentación (GUI):**
 - Manejo de pantallas por rol
 - Tablas dinámicas
 - Entradas por diálogo
 - Generación de QR
 - Impresión de tickets
- **Capa de pruebas (JUnit):**
 - Validación de compra
 - Validación de estados del ticket
 - Registro de impresión
 - Asociación de evento/localidad

5.2 Flujo principal (Cliente)

1. Login
2. Menú: Comprar tickets / Mis tickets
3. Compra: selección → localidad → confirmación
4. Mis tickets: tabla con estado dinámico
5. Imprimir ticket: registra fecha/hora
6. Generar QR: crea archivo PNG del código QR

5.3 Generación de QR

Utiliza ZXing:

- Convierte el ID del ticket en un código
 - Genera un PNG en /data/qr/tickete_<ID>.png
 - Muestra confirmación al usuario
-

6. Requerimientos

Funcionales

- Inicio de sesión para todos los roles.
- Compra de tickets desde interfaz gráfica.
- Impresión de tickets (cambia su estado).
- Generación y almacenamiento de códigos QR.
- Listado dinámico de tickets adquiridos.
- Creación y aprobación de venues.
- Creación de eventos por organizadores.
- Persistencia automática en JSON.

No funcionales

- Bajo acoplamiento entre GUI y dominio.
 - Respuestas inmediatas de la interfaz.
 - Manejo claro de errores en ventana emergente.
 - Compatibilidad con ZXing sin acoplar dominio a la librería.
 - Código legible y modular.
-

7. Historias de Usuario

7.1 Cliente compra un tickete desde la GUI

Actor: Cliente

Descripción: Comprará un tickete seleccionando evento/localidad.

Entradas: Evento, localidad, cantidad.

Resultado esperado: Tickete creado y visible en “Mis tickets”.

7.2 Cliente imprime un tickete

Descripción: El cliente selecciona un tickete en la tabla y lo imprime.

Resultado esperado: fechaImpresion registrada y estado actualizado.

7.3 Cliente genera el QR

Descripción: El cliente presiona “Generar QR” sobre un tickete impreso.

Resultado esperado: PNG creado con éxito.

7.4 Organizador crea eventos

Descripción: Registro completo de un evento aprobado con localidades.

7.5 Administrador aprueba venues

Descripción: Lista venues no aprobados, elige uno y lo aprueba.

8. Persistencia

- El sistema utiliza JsonStore para mantener usuarios, eventos y tickets entre ejecuciones.
- la impresión y QR no se guardan (solo su estado, no imágenes).
- La persistencia se activa en cada acción relevante del usuario.
- Los datos siempre se restauran al abrir nuevamente el programa.

9. Conclusiones

La implementación final de Boletamaster provee una experiencia completa y visual para todos los tipos de usuario. La integración de QR, impresión de tiquetes y tablas dinámicas representa una evolución significativa del proyecto en comparación con la versión por consola.

Además, la arquitectura modular y la persistencia estable permiten extender el sistema hacia nuevas funcionalidades en futuras entregas, manteniendo claridad, legibilidad y coherencia con los principios de diseño orientado a objetos. Lamentablemente en la entrega final no se logró correr todo de manera correcta. Se espera que este documento pueda guiar y dar idea de lo que se pensaba lograr con el código que se desarrolló, gracias.
