

UNIVERSIDAD PRIVADA DOMINGO SAVIO

FACULTAD DE INGENIERÍA

SISTEMAS Y REDES



UNIVERSIDAD PRIVADA
DOMINGO SAVIO

TRABAJO:

Proyecto

Grupo 6(S):

Roque Fernando Velasquez Velasquez

DOCENTE:

Ing. Juan Carlos Huanto V.

TARIJA, NOVIEMBRE 2025

ÍNDICE GENERAL

1. Introducción	1
1.1 Contexto del proyecto	1
1.2 Problemática detectada	1
2. Propósito del Sistema	2
3. Objetivo General	2
4. Objetivos Específicos	3
5. Justificación del Proyecto	4
5.1 Necesidad de la solución	4
5.2 Beneficios técnicos y funcionales	4
6. Alcance del Sistema	5
7. Arquitectura y Estructura del Proyecto	6
8. Módulos Funcionales	7
8.1 Gestión de Eventos	7
8.2 Gestión de Artistas	7
8.3 Gestión de Ventas y Entradas	7
9. Tecnologías Utilizadas	8
10. Resultados Esperados	9
11. Conclusiones	10
12. Proyecciones Futuras	11

INFORME GENERAL DEL PROYECTO

Sistema de Gestión de Eventos Culturales

1. Introducción

El proyecto Evento_Cultural es una aplicación web desarrollada con ASP.NET Core MVC y Entity Framework Core, diseñada para gestionar eventos culturales. Permite administrar artistas, eventos, entradas, ventas, participaciones y ganadores, además de generar reportes y estadísticas. Está orientada a facilitar el control completo de actividades relacionadas con la organización de un evento cultural, integrando múltiples módulos, lógica de negocio y visualización atractiva.

La aplicación se ejecuta sobre un servidor web con conexión a una base de datos PostgreSQL utilizando el ORM Entity Framework Core. Sigue la arquitectura MVC (Modelo-Vista-Controlador), separando claramente la lógica de presentación, la lógica de negocio y el acceso a datos, lo cual mejora la escalabilidad y mantenibilidad del sistema.

2. Objetivos del Proyecto

Objetivo general:

Desarrollar un sistema completo que permita gestionar de manera eficiente los eventos culturales, incluyendo artistas, entradas, ventas, participantes, ganadores y estadísticas generales con visualización.

Objetivos específicos:

- Registrar y administrar eventos con datos como nombre, fecha, lugar, estado y costo.
- Gestionar artistas participantes y su rendimiento.
- Registrar entradas y ventas.
- Controlar la asistencia y participación.
- Generar reportes e indicadores mediante servicios.
- Mostrar estadísticas en un dashboard gráfico.

3. Arquitectura del Sistema

El sistema sigue la arquitectura MVC con una organización basada en capas:

1. **Models (Modelo):** Gestión de datos, entidades y relaciones.

2. **Views (Vista):** Interfaz de usuario, páginas Razor.
3. **Controllers (Controlador):** Gestión del flujo, acciones HTTP.
4. **Services (Servicios):** Lógica de negocio y consultas avanzadas.
5. **DTOs:** Objetos para transferir datos personalizados.

4. Estructura del Proyecto

Evento_Cultural/

```

|
├── Controllers/      → Controladores MVC
├── DTOs/             → Objetos de transferencia de datos
├── Models/           → Entidades, DbContext y estructura de
datos
├── Services/         → Lógica de negocio independiente
├── Views/            → Vistas para UI
├── wwwroot/          → Archivos estáticos (imágenes, css,
js)
├── Program.cs        → Configuración general
└── appsettings.json  → Configuración de conexión a base de
datos

```

5. Módulos Principales del Sistema

5.1 Gestión de Eventos

Permite crear, editar, eliminar y consultar eventos. Tiene atributos como:

- Id, Nombre, Fecha, Lugar, Estado, Precio.

5.2 Gestión de Artistas

Controla los datos del artista y su rendimiento:

- Nombre artístico, especialidad, popularidad.

5.3 Gestión de Entradas

Administra la venta de entradas, precios, tipo, evento asociado y disponibilidad.

5.4 Gestión de Ventas

Registra las transacciones realizadas:

- Evento, cantidad vendida, monto total.

5.5 Participaciones y Ganadores

Administra a los participantes de concursos dentro del evento cultural y registra ganadores.

5.6 Dashboard (Panel de Estadísticas)

Muestra resúmenes mediante consultas a los servicios:

- Evento con más ventas.
- Artista más popular.
- Ingresos generados por evento.
- Total de asistentes.

6. Clases Principales del Modelo

El sistema abarca el registro, control y administración de eventos culturales, incluyendo la gestión de artistas, participantes, entradas, ventas y ganadores. Permite el almacenamiento de datos, consulta de información y generación de reportes básicos. Se enfoca en el funcionamiento interno del evento y no incluye pasarela de pagos externa, ni integración con sistemas de transmisión en vivo.

Clase	Descripción
Evento	Información general del evento.
Artista	Datos del artista participante.
Entrada	Datos de las entradas vendidas y disponibles.
Venta	Registro de ventas ligadas a cada evento.
Participacion	Relación entre artista y evento.
Ganador	Ganadores de concursos del evento.
Reporte	Información generada para reportes específicos.

El archivo **ApplicationDbContext.cs** administra las entidades y genera automáticamente tablas en PostgreSQL.

7. Controladores

El proyecto está desarrollado en ASP.NET Core MVC bajo una arquitectura de capas, separando los componentes en:

- **Controllers:** Gestionan solicitudes del usuario y coordinan la comunicación entre vista y modelo.
- **Models:** Representan las entidades del sistema como Evento, Artista, Entrada, Venta, Participación y Ganador.
- **Services:** Implementan la lógica de negocio, cálculos, validaciones y reportes.
- **Views:** Interfaz presentada al usuario, estructurada en carpetas por módulo, utilizando Razor Pages.

Se emplea Entity Framework Core para el manejo de base de datos PostgreSQL.

Controlador	Funcionalidad
EventosController	Gestión CRUD de eventos.
ArtistasController	Administración de artistas.
EntradasController	Control de entradas.
VentasController	Registro de ventas.
ParticipacionesController	Participaciones de artistas en eventos.
GanadoresController	Registro de ganadores.
DashboardController	Vista principal con estadísticas.

8. Servicios

8.1 Gestión de Eventos: Registro, edición, seguimiento del estado (Programado, En Curso, Finalizado), precio, lugar y fecha.

8.2 Gestión de Artistas: Registro de artistas, detalles, participación y rendimiento.

8.3 Gestión de Entradas y Ventas: Registro de entradas disponibles, cantidad vendida, cálculo de total, fecha de venta y cliente.

8.4 Participaciones y Ganadores: Asociación entre evento, artistas y resultados, selección de ganadores y categorías.

8.5 Dashboard estadístico: Visualización de datos resumidos sobre eventos, ventas y rendimiento.

Servicio	Función principal
EventoService	Obtención de detalles avanzados de eventos.
DashboardService	Generación de estadísticas para dashboard.
VentaService	Cálculo de ingresos, ventas y totales.
ReporteService	Creación de reportes personalizados.
ArtistaService	Métricas de rendimiento de artistas.

Este enfoque permite mantener controladores limpios y reutilizar lógica.

9. Vistas

. Tecnologías Utilizadas

- **Lenguaje principal:** C#
- **Framework:** ASP.NET Core MVC
- **Base de datos:** PostgreSQL con Entity Framework Core
- **Frontend:** Razor Pages, Tailwind CSS, HTML5
- **IDE:** Visual Studio / Visual Studio Code
- **Servidor local:** Kestrel
- **Control de versiones:** Git y GitHub

Carpeta	Función
Eventos	Formularios y vistas CRUD para eventos.
Artistas	Vista de datos de artistas.
Ventas	Registro y detalle de ventas.
Entradas	Gestión de entradas.
Shared	Layout general, validaciones, estilos.
Dashboard	Panel de estadísticas central.

El archivo `_Layout.cshtml` define el menú, encabezado y diseño general.

10. Persistencia en Base de Datos

Se utiliza PostgreSQL con Entity Framework Core. El contexto `ApplicationDbContext` permite realizar migraciones automáticas y consultas mediante LINQ.

El archivo `appsettings.json` contiene la cadena de conexión:

```
"ConnectionStrings": {
  "DefaultConnection":
  "Host=localhost;Database=EventoCultural;Username=postgres;Password=...;"
}
```

11. Flujo General del Sistema

1. El usuario ingresa al Dashboard.
2. Observa estadísticas generales del sistema.
3. Puede navegar hacia eventos, artistas, ventas o participación.
4. Puede registrar, editar o consultar registros.
5. Los datos se guardan automáticamente en PostgreSQL.
6. Los servicios procesan los datos y generan informes.

12. Análisis Final

El sistema cumple con los principios de separación de responsabilidades, modularidad, reutilización de código y una arquitectura escalable. Permite fácilmente extenderse con nuevos módulos, como reportes PDF, autenticación, gráficos interactivos o integración con APIs externas.

13. Conclusión

El proyecto `Evento_Cultural` es una aplicación web robusta, correctamente organizada y basada en buenas prácticas de diseño. Utiliza la arquitectura MVC, separación clara de capas y servicios independientes, lo que permite mantener el código ordenado, reutilizable y escalable. Su estructura permite agregar nuevas funcionalidades sin modificar el núcleo del sistema.