MANUAL TÉCNICO Y DE OPERACIÓN DEL SISTEMA EVENTSOFT

Sistema de Gestión de Eventos Académicos y Corporativos

Control de Versiones

Versión:	Fecha:	Descripción:	Autores:
1. 0	6/9/2025	**Versión inicial técnico**	**Santiago Alzate** **Santiago Molano** **Sebastián Perdomo** **Alejandro Quiceno**

TABLA DE CONTENIDO

1.OBJETIVO

2 ALCANCE

3.TÉRMINOS Y DEFINICIONES

4. ¿QUÉ ES EL MANUAL TÉCNICO Y DE OPERACIÓN DEL SISTEMA?

5. INTRODUCCIÓN

6.DESCRIPCIÓN DEL SISTEMA DE INFORMACIÓN DESARROLLADO

7.DISEÑO TÉCNICO DEL SISTEMA DE INFORMACIÓN

8.DESPLIEGUE Y CONFIGURACIÓN DE COMPONENTES

9.SOLUCIÓN DE PROBLEMAS

1. OBJETIVO

El objetivo de este documento es brindar una guía técnica completa para la operación, administración y mantenimiento del Sistema EventSoft, ilustrando sobre la definición, diseño, organización y estructura del sistema al personal encargado de mantener la prestación del servicio, incluyendo desarrolladores, arquitectos, administradores de sistemas e ingenieros de soporte.

2. ALCANCE

Este documento describe el contenido técnico completo del sistema EventSoft, incluyendo su arquitectura, componentes, procedimientos de instalación, configuración, despliegue y operación, así como la resolución de problemas técnicos más comunes.

3. TÉRMINOS Y DEFINICIONES

Django: Framework de desarrollo web de alto nivel escrito en Python que fomenta el desarrollo rápido y el diseño limpio y pragmático.

SQLite: Sistema de gestión de bases de datos relacional contenido en una biblioteca de C.

Bootstrap: Framework de CSS para el desarrollo de interfaces web responsivas y mobilefirst.

WeasyPrint: Biblioteca de Python para generar documentos PDF desde HTML y CSS.

MVC (Model-View-Controller): Patrón de arquitectura de software que separa los datos y la lógica de negocio de la interfaz de usuario.

ORM (Object-Relational Mapping): Técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y la base de datos.

4. ¿QUÉ ES EL MANUAL TÉCNICO Y DE OPERACIÓN DEL SISTEMA?

El manual técnico del Sistema EventSoft tiene como propósito ilustrar sobre la definición, diseño, organización y estructura del sistema al personal encargado de mantener la prestación del servicio de gestión de eventos académicos y corporativos.

Este documento es fundamental para garantizar la continuidad operativa del sistema, facilitar el mantenimiento, y servir como guía para futuras actualizaciones o modificaciones del sistema.

5. INTRODUCCIÓN

EventSoft es un sistema web desarrollado para la gestión integral de eventos académicos y corporativos. El sistema permite la administración completa del ciclo de vida de un evento, desde su creación hasta la generación de certificados de participación.

El sistema fue desarrollado utilizando el framework Django de Python, siguiendo las mejores prácticas de desarrollo web y patrones de diseño establecidos.

6. DESCRIPCIÓN DEL SISTEMA DE INFORMACIÓN DESARROLLADO

6.1 Índice del Contenido

El Sistema EventSoft está compuesto por los siguientes módulos principales:

- **Módulo de Administración General** (`app_admin`)
- **Módulo de Administradores de Eventos** (`app_administradores`)
- **Módulo de Gestión de Eventos** (`app_eventos`)
- **Módulo de Participantes** (`app_participantes`)
- **Módulo de Evaluadores** (`app_evaluadores`)
- **Módulo de Asistentes** (`app_asistentes`)
- **Módulo de Usuarios** (`app_usuarios`)
- **Módulo de Áreas** (`app_areas`)

6.2 Introducción

EventSoft es una plataforma web integral que automatiza y digitaliza la gestión de eventos académicos y corporativos. El sistema permite:

- Creación y gestión de eventos
- Registro y administración de participantes, evaluadores y asistentes
- Generación automática de códigos de invitación
- Sistema de validación mediante códigos QR
- Generación de certificados personalizables
- Gestión de usuarios con diferentes roles y permisos
- Notificaciones automáticas por correo electrónico

6.3 Objetivos del Sistema

Objetivo General:

Desarrollar un sistema web que permita la gestión integral de eventos académicos y corporativos, automatizando los procesos de registro, validación y certificación de participantes.

Objetivos Específicos:

1. Automatizar el proceso de registro de participantes en eventos
2. Implementar un sistema de validación mediante códigos QR
3. Generar certificados personalizables en formato PDF
4. Proporcionar interfaces de usuario intuitivas para diferentes roles
5. Garantizar la seguridad y integridad de los datos
6. Facilitar la gestión administrativa de eventos

7. **DISEÑO TÉCNICO DEL SISTEMA DE INFORMACIÓN**

7.1 Esquema o Modelo de Requerimientos

Requerimientos Funcionales Principales:

RF001 - Gestión de Eventos:

- Crear, editar, eliminar y consultar eventos
- Definir fechas, ubicaciones y configuraciones específicas

RF002 - Gestión de Usuarios:

- Registro y autenticación de usuarios
- Asignación de roles (Admin, Administrador de Evento, Participante, Evaluador, Asistente)

RF003 - Sistema de Inscripciones:

- Registro de participantes mediante códigos de invitación
- Validación automática de datos
- Confirmación por correo electrónico

RF004 - Generación de Certificados:

- Plantillas personalizables (Elegante, Moderno, Clásico)
- Generación automática en formato PDF
- Campos dinámicos reemplazables

RF005 - Sistema de Validación QR:

- Generación de códigos QR únicos
- Validación de asistencia
- Registro de participación

Requerimientos No Funcionales:

RNF001 - Seguridad:

- Autenticación y autorización basada en roles
- Protección contra ataques CSRF
- Validación de datos de entrada

RNF002 - Usabilidad:

- Interfaz responsive compatible con dispositivos móviles
- Diseño intuitivo y moderno
- Tiempo de respuesta inferior a 3 segundos

RNF003 - Compatibilidad:

- Compatible con navegadores modernos (Chrome, Firefox, Safari, Edge)
- Soporte para dispositivos móviles y tablets

7.2 Software Base del Sistema y Prerequisitos

Requerimientos Mínimos de Hardware:

- **CPU:** 2 cores, 2.0 GHz
- **RAM:** 4 GB
- **Disco Duro:** 20 GB de espacio libre
- **Red:** Conexión a Internet

Requerimientos Recomendados de Hardware:

- **CPU:** 4 cores, 2.5 GHz o superior
- **RAM:** 8 GB o superior
- **Disco Duro:** 50 GB de espacio libre SSD
- **Red:** Conexión a Internet estable

Requerimientos de Software:

Sistema Operativo:

- Windows 10/11, macOS 10.14+, o Linux (Ubuntu 18.04+)

Software Base:

- **Python:** 3.8 o superior
- **Base de Datos:** SQLite 3.x (incluido con Python)
- **Navegadores Soportados:**
- Google Chrome 90+
- Mozilla Firefox 88+
- Safari 14+
- Microsoft Edge 90+

7.3 Componentes y Estándares

Frameworks y Librerías Principales:

Backend:

- **Django 4.2:** Framework web principal
- **Django ORM:** Mapeo objeto-relacional
- **WeasyPrint:** Generación de PDFs
- **Pillow:** Procesamiento de imágenes
- **qrcode:** Generación de códigos QR

Frontend:

- **Bootstrap 5.3:** Framework CSS
- **Bootstrap Icons:** Iconografía
- **jQuery:** Manipulación DOM y AJAX

- **HTML5/CSS3:** Estándares web

Estándares de Codificación:

- **PEP 8:** Estilo de código Python

- **Django Best Practices:** Convenciones del framework - **Responsive Design:** Principios de diseño adaptativo

- **RESTful URLs:** Estructura de URLs semánticas

Patrones de Diseño Implementados:

- **MVC (Model-View-Controller):** Arquitectura principal

- **Template Method:** Herencia de plantillas Django

- **Decorator Pattern:** Middleware y decoradores de vistas

- **Observer Pattern:** Señales de Django

7.4 Modelo de Datos

Estructura de Base de Datos

Entidades Principales:

1. **Usuario (User):**

- Extiende el modelo de usuario de Django

- Campos: username, email, first_name, last_name, documento, telefono

2. **Evento (Evento):**

- eve_id (PK)

- eve_nombre

- eve_descripcion

- eve_fecha_inicio

- eve_fecha_fin

- eve_ciudad

- eve_lugar

- eve_estado

- eve_costo

3. **Participante (Participante):**

- par_id (PK)

- usuario (FK)

- par_profesion

- par_institucion

4. **EventoParticipante (EventoParticipante):**

- Relación many-to-many entre Evento y Participante

- par_eve_estado

- par_eve_fecha_inscripcion

5. **ConfiguracionCertificado (ConfiguracionCertificado):**

- evento (FK)

- tipo

- titulo

- cuerpo

- plantilla

- logo

- firma

Diagrama Entidad-Relación

```

[Usuario] 1:N [Participante] N:M [Evento]

[Usuario] 1:N [Evaluador] N:M [Evento]

[Usuario] 1:N [Asistente] N:M [Evento]

[Usuario] 1:N [Administrador] N:M [Evento]

[Evento] 1:N [ConfiguracionCertificado]

[Evento] 1:N [CodigoInvitacion]

**7.5 Funcionalidad y Servicios Ofrecidos**

**Módulos del Sistema:**

**1. Gestión de Eventos:**

- Creación de eventos con información detallada
- Configuración de fechas y ubicaciones
- Gestión del estado del evento (Activo, Finalizado, Cancelado)

**2. Sistema de Usuarios:**

- Registro y autenticación
- Gestión de perfiles
- Asignación de roles específicos

**3. Inscripciones:**

- Registro mediante códigos de invitación
- Validación automática de datos
- Confirmación por correo electrónico

**4. Generación de Certificados:**

- Plantillas personalizables
```

- Campos dinámicos

- Exportación a PDF de alta calidad

5. Validación QR:

- Generación de códigos únicos

- Escaneo para validación de asistencia

- Registro de participación en tiempo real

8. DESPLIEGUE Y CONFIGURACIÓN DE COMPONENTES

8.1 Organización de Componentes

Estructura del Proyecto: eventsoft/

```
├── manage.py
├── pr_eventsoft/
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   ├── wsgi.py
│   └── asgi.py
├── app_admin/
│   ├── models.py
│   ├── views.py
│   ├── urls.py
│   └── templates/
├── app_eventos/
│   ├── models.py
```

```
│   ├── views.py
│   ├── urls.py
│   └── templates/
├── app_usuarios/
│   ├── models.py
│   ├── views.py
│   ├── middleware.py
│   └── permisos.py
├── static/
│   ├── css/
│   ├── js/
│   └── img/
├── media/
│   ├── logos/
│   ├── firmas/
│   └── documentos/
└── templates/
    └── base.html
```

Diagrama de Componentes:

[Interfaz Web] -> [Django Views] -> [Models/ORM] -> [SQLite DB]

[Interfaz Web] -> [Static Files] -> [CSS/JS/Images]

[Django Views] -> [WeasyPrint] -> [PDF Generator]

[Django Views] -> [Email Backend] -> [SMTP Server]

8.2 Instalación

Pre-requisitos de Instalación:

1. **Python 3.8+** instalado en el sistema
2. **pip** (gestor de paquetes de Python)
3. **virtualenv** (recomendado para aislamiento)

Pasos de Instalación:

1. Clonar el repositorio:

```bash
git clone https://github.com/usuario/eventsoft.git cd
eventsoft
```

```

2. **Crear entorno virtual:**

```bash python -m venv venv #
Windows venv\Scripts\activate
# Linux/macOS source
venv/bin/activate
```

3. **Instalar dependencias:** ```bash pip install -r requirements.txt
```

4. **Configurar base de datos:** ```bash python manage.py makemigrations
python manage.py migrate
```

5. **Crear superusuario:**

```bash python manage.py
createsuperuser
```

6. **Recopilar archivos estáticos:** ```bash python manage.py collectstatic

Requirements.txt:

Django==4.2.0
```

```

Pillow==9.5.0 WeasyPrint==59.0

qrcode==7.4.2

**8.3 Configuración**

**Variables de Configuración (settings.py):**

**Base de Datos:**

```python
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

Configuración de Email:

```python
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 587
EMAIL_USE_TLS = True
EMAIL_HOST_USER = 'your-email@gmail.com'
EMAIL_HOST_PASSWORD = 'your-app-password'
```

Archivos Media:
```

```

```python
MEDIA_URL = '/media/'
MEDIA_ROOT = BASE_DIR / 'media'
```

Configuración de Seguridad:

```python
SECRET_KEY = 'your-secret-key-here'
DEBUG = False  # En producción
ALLOWED_HOSTS = ['localhost', '127.0.0.1', 'your-domain.com']
```

Configuración de Roles y Permisos:

El sistema utiliza el sistema de permisos nativo de Django con grupos personalizados:

- **Administradores Generales**
- **Administradores de Eventos**
- **Participantes**
- **Evaluadores**
- **Asistentes**

8.4 Despliegue

Despliegue Local (Desarrollo):

```bash
python manage.py runserver 0.0.0.0:8000
```

Despliegue en Producción:
```

```

**1. Configurar servidor web (Apache/Nginx):**

```apache
<VirtualHost *:80>
    ServerName your-domain.com
    WSGIScriptAlias / /path/to/eventsoft/pr_eventsoft/wsgi.py

    <Directory /path/to/eventsoft>
        Require all granted
    </Directory>

    Alias /static /path/to/eventsoft/static
    Alias /media /path/to/eventsoft/media
</VirtualHost>
```

2. Configurar variables de entorno: ```bash export
SECRET_KEY="production-secret-key" export DEBUG=False
export ALLOWED_HOSTS="your-domain.com"
```

3. Configurar HTTPS:

```bash
# Instalar certificado SSL sudo certbot --
apache -d your-domain.com **Diagrama de**
**Despliegue:**
```

```

```
[Internet] -> [Load Balancer] -> [Web Server (Apache/Nginx)]
                                    |
                                [Django App] -> [SQLite DB]
                                    |
                                [Static Files]
                                    |
                                [Media Files]
```

## 9. RESOLUCIÓN DE PROBLEMAS

### 9.1 Errores Técnicos Más Comunes y Su Solución

#### Error 1: "ModuleNotFoundError: No module named 'django'"

**Síntomas:**

- Error al ejecutar comandos de Django
- Sistema no encuentra el módulo Django

**Posibles Causas:**

- Entorno virtual no activado
- Django no instalado

**Diagnóstico:**

```bash

python -c "import django; print(django.VERSION)"

```

Solución:

```bash

# Activar entorno virtual source

venv/bin/activate  # Linux/macOS

venv\Scripts\activate     # Windows

# Instalar Django pip

install Django==4.2.0

```

Error 2: "django.db.utils.OperationalError: no such table"

Síntomas:

- Error al acceder a páginas que requieren base de datos

- Mensaje de tabla no encontrada

Posibles Causas:

- Migraciones no aplicadas

- Base de datos corrupta

Diagnóstico: ```bash

python manage.py showmigrations

```

**Solución:**

```bash
```

python manage.py makemigrations python

manage.py migrate

```

**Error 3: "WeasyPrint: Failed to load image"**

**Síntomas:**

- Error al generar certificados PDF

- Imágenes no se muestran en PDFs

**Posibles Causas:**

- Ruta de imagen incorrecta

- Archivo de imagen corrupto

- Permisos de archivo insuficientes

**Diagnóstico:**

- Verificar que los archivos existan en la ruta especificada

- Comprobar permisos de lectura

**Solución:**

```bash

# Verificar permisos

chmod 644 media/logos/*

chmod 644 media/firmas/*

# Verificar configuración MEDIA_ROOT python

manage.py shell

>>> from django.conf import settings

>>> print(settings.MEDIA_ROOT)

```

Error 4: "CSRF verification failed"

Síntomas:

- Error 403 Forbidden en formularios

- Mensaje de verificación CSRF fallida

Posibles Causas:

- Token CSRF ausente o inválido

- Configuración de middleware incorrecta

Diagnóstico:

- Verificar que {% csrf_token %} esté en formularios

- Comprobar middleware en settings.py

Solución:

```python
# En templates, asegurar que todos los formularios tengan:
<form method="post">
    {% csrf_token %}
    <!-- campos del formulario -->
</form>

# En settings.py, verificar middleware:
MIDDLEWARE = [
    'django.middleware.csrf.CsrfViewMiddleware',
# otros middleware...
```
```

]

```

**Error 5: "Permission denied" al subir archivos**

**Síntomas:**

- Error al subir logos o firmas

- Archivos no se guardan correctamente

**Posibles Causas:**

- Permisos insuficientes en directorio media

- Configuración incorrecta de MEDIA_ROOT

**Diagnóstico:**

```bash ls -la

media/

```

Solución:

```bash

# Crear directorios si no existen mkdir

-p media/logos media/firmas

# Configurar permisos

chmod 755 media/

chmod 755 media/logos/

chmod 755 media/firmas/

```

Error 6: "Correos no se envían"
```

Síntomas:

- Usuarios no reciben emails de confirmación

- No hay errores visibles en la aplicación

Posibles Causas:

- Configuración SMTP incorrecta

- Credenciales de email inválidas

- Firewall bloqueando puerto 587

Diagnóstico: ```python

python manage.py shell

>>> from django.core.mail import send_mail

>>> send_mail('Test', 'Message', 'from@email.com', ['to@email.com'])

```

**Solución:**

```python

# Verificar configuración en settings.py

EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'

EMAIL_HOST = 'smtp.gmail.com'

EMAIL_PORT = 587

EMAIL_USE_TLS = True

EMAIL_HOST_USER = 'your-email@gmail.com'

EMAIL_HOST_PASSWORD = 'your-app-password'  # App password, no regular password

# Para Gmail, generar contraseña de aplicación:

# 1. Activar 2FA en cuenta Google

# 2. Generar contraseña de aplicación

# 3. Usar esa contraseña en EMAIL_HOST_PASSWORD
```

9.2 Logs y Monitoreo

Configuración de Logging:

```python
# En settings.py
LOGGING = {
    'version': 1,
    'disable_existing_loggers': False,
    'handlers': {
        'file': {
            'level': 'INFO',
            'class': 'logging.FileHandler',
            'filename': 'eventsoft.log',
        },
    },
    'loggers': {
        'django': {
            'handlers': ['file'],
            'level': 'INFO',
            'propagate': True,
        },
    },
}
```
```

Comandos Útiles para Diagnóstico:

```bash
# Ver logs en tiempo real
tail -f eventsoft.log

# Verificar estado de la aplicación python
manage.py check

# Verificar configuración python
manage.py diffsettings

# Verificar base de datos python
manage.py dbshell
```

Documento elaborado siguiendo los lineamientos del Marco de Referencia de Arquitectura TI de MinTIC y las mejores prácticas de documentación técnica.