**Sistema CEVAZ Editor**



**Manual de Documentación Final**

**Usuarios administradores**

Este documento debe ser entregado a todos los administradores para garantizar el correcto uso, mantenimiento y seguridad del sistema CEVAZ Bot Editor.

Índice

**Contenido**

[1. ESTRUCTURA DEL PROYECTO 3](#_Toc201582437)

[Árbol de carpetas 3](#_Toc201582438)

[Diagrama de flujo del sistema 4](#_Toc201582439)

[Entidades del sistema 4](#_Toc201582440)

[Diagrama entidad-relación 5](#_Toc201582441)

[2. ARCHIVOS PRINCIPALES 5](#_Toc201582442)

[3. Flujo de uso para administradores 8](#_Toc201582443)

[4. Seguridad y buenas prácticas 8](#_Toc201582444)

[5. Mantenimiento y recuperación 9](#_Toc201582445)

[6. Personalización 9](#_Toc201582446)

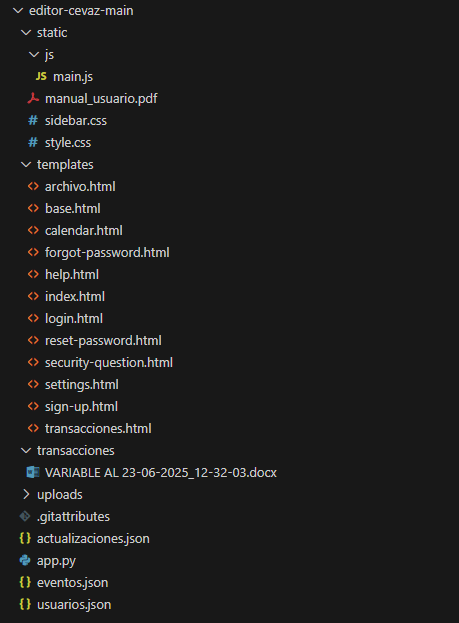
[7. Notificaciones y correos 9](#_Toc201582447)

[8. Recomendaciones finales 9](#_Toc201582448)

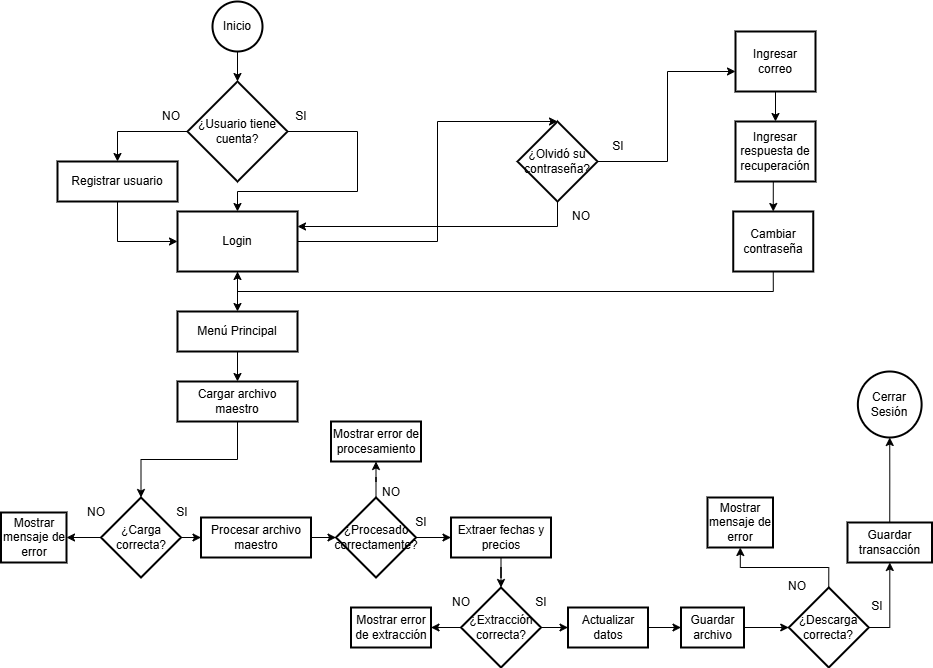
[9. Contacto y soporte 10](#_Toc201582449)

# ESTRUCTURA DEL PROYECTO

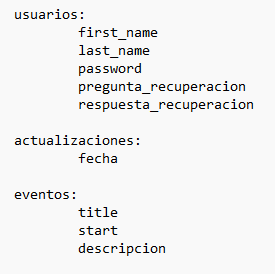
## árbol de carpetas



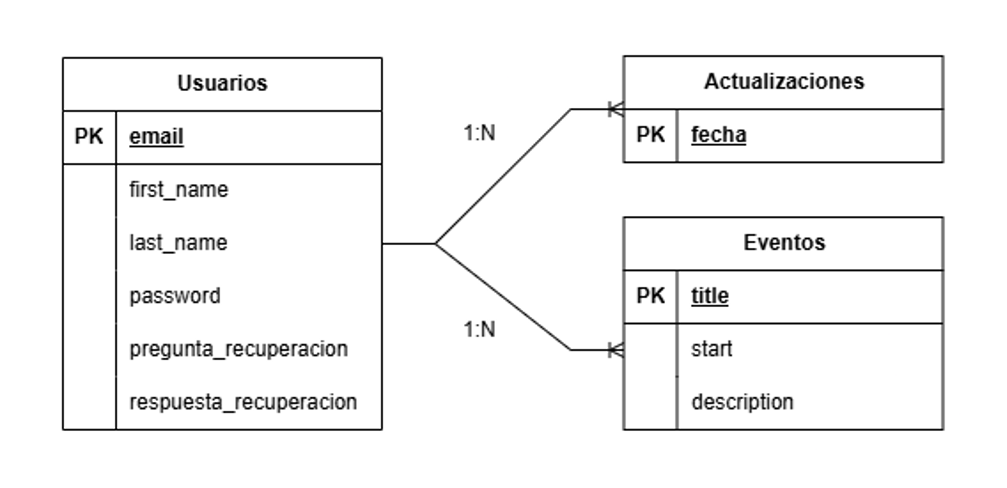
## diagrama de flujo del sistema



## entidades del sistema



## diagrama entidad-relación



# 2. ARCHIVOS PRINCIPALES

a) ***app.py***

* **Tipo:** Código fuente principal (Python/Flask)
* **Función:**

Orquesta toda la lógica del sistema:

* Autenticación y gestión de usuarios
* Procesamiento y actualización de archivos .docx
* Gestión de eventos y calendario
* Envío de correos automáticos
* Manejo de sesiones y seguridad
* Rutas para todas las vistas y acciones del sistema.

**Funcionalidades destacadas:**

* Login, registro y recuperación de contraseña: Incluye validación de credenciales, preguntas de seguridad y restablecimiento seguro.
* Procesamiento de archivos: Permite subir, analizar y actualizar archivos .docx, detectando fechas y precios automáticamente.
* Historial de transacciones: Guarda cada versión generada del archivo en la carpeta transacciones.
* Calendario de eventos: Visualiza eventos y actualizaciones, permite crear nuevos eventos y envía notificaciones por correo.
* Configuración de usuario: Cambia correo, contraseña y pregunta de recuperación desde la interfaz.
* Envío de correos: Usa Flask-Mail para notificar eventos y enviar archivos actualizados.
* Tareas programadas: Usa APScheduler para enviar recordatorios automáticos de eventos.

b) ***usuarios.json***

* **Tipo:** Archivo de datos (JSON)
* **Función:**

Almacena la lista de usuarios registrados, con los siguientes campos:

* email: Correo electrónico (único)
* first\_name, last\_name: Nombre y apellido
* password: Contraseña cifrada (scrypt)
* pregunta\_recuperacion: Pregunta de seguridad
* respuesta\_recuperacion: Respuesta a la pregunta

**Recomendaciones:**

No editar manualmente salvo emergencia.

Realizar respaldos periódicos.

c) ***eventos.json***

* **Tipo:** Archivo de datos (JSON)
* **Función:**

Lista de eventos personalizados del calendario. Cada evento contiene:

* title: Título del evento
* start: Fecha (YYYY-MM-DD)
* description: Descripción
* creador: Correo del usuario que creó el evento

**Recomendaciones:**

Puede editarse manualmente para mantenimiento, pero preferible usar la interfaz.

d) ***actualizaciones.json***

* **Tipo:** Archivo de datos (JSON)
* **Función:**

Guarda la fecha de cada actualización realizada sobre el archivo maestro. Se utiliza para mostrar el historial de cambios en el calendario.

e) ***uploads/***

* **Tipo:** Carpeta
* **Función:**

Almacena los archivos .docx subidos por los usuarios para ser procesados.

f) ***transacciones/***

* **Tipo:** Carpeta
* **Función:**

Guarda todas las versiones generadas del archivo maestro tras cada actualización. Cada archivo lleva la fecha y hora en el nombre para fácil identificación.

g) ***templates/***

* **Tipo:** Carpeta
* **Función:**

Contiene todas las vistas del sistema.

* login.html: Formulario de acceso
* sign-up.html: Registro de usuario
* forgot-password.html: Recuperación de contraseña
* security-question.html: Validación de pregunta de seguridad
* reset-password.html: Restablecimiento de contraseña
* archivo.html: Subida y edición de archivos
* calendar.html: Calendario de eventos y actualizaciones
* transacciones.html: Historial de archivos generados
* settings.html: Configuración de usuario
* help.html: Ayuda y documentación
* base.html: Plantilla base para heredar estilos y estructura

h) ***static/***

* **Tipo: Carpeta de archivos estáticos**
* **Función:**

Contiene CSS, JS e imágenes para la interfaz.

* style.css: Estilos generales
* sidebar.css: Estilos para el menú lateral y dashboard
* js/main.js: Scripts para interacción de la interfaz

# 3. flujo de uso para administradores

* Acceso: Ingresar con usuario y contraseña. Si olvida la contraseña, puede recuperarla respondiendo la pregunta de seguridad.
* Subida de archivos: Desde la sección "Archivo", subir el documento .docx a actualizar. El sistema detecta automáticamente fechas y precios.
* Edición de datos: Modificar fechas y precios desde la interfaz antes de guardar los cambios.
* Actualización y descarga: Al guardar, se genera una nueva versión en transacciones, se registra la actualización y se puede descargar el archivo actualizado.
* Calendario: Visualizar todas las actualizaciones y eventos. Crear nuevos eventos para notificar a otros usuarios.
* Historial: Acceder a todas las versiones anteriores desde "Transacciones".
* Configuración: Cambiar correo, contraseña o pregunta de seguridad desde "Configuración".
* Ayuda: Consultar la sección de ayuda para dudas frecuentes.

# 4. Seguridad y buenas prácticas

* Contraseñas cifradas: Nunca se almacenan en texto plano.
* Sesiones seguras: Solo usuarios autenticados pueden acceder a funciones críticas.
* Validación de formularios: Todos los datos se validan antes de procesarse.
* Respaldo de archivos: Realizar copias periódicas de los archivos JSON y carpetas de archivos.
* No compartir credenciales: Mantener en secreto la clave de correo y la clave secreta de Flask.

# 5. mantenimiento y recuperación

* Restaurar usuarios: Reemplazar usuarios.json por un respaldo si es necesario.
* Restaurar eventos: Reemplazar eventos.json por un respaldo.
* Restaurar archivos: Descargar cualquier versión desde la sección "Transacciones".
* Actualizar dependencias: Usar pip install -r requirements.txt si se agregan nuevas librerías.

# 6. personalización

* Plantillas HTML: Puede modificar los archivos en templates para adaptar la imagen institucional.
* Estilos CSS: Editar style.css y sidebar.css para cambiar colores, fuentes y disposición.
* Preguntas de seguridad: Puede agregar más preguntas en la sección correspondiente del código o la interfaz.

# 7. notificaciones y correos

* Envío automático: El sistema envía correos al crear eventos y al actualizar archivos.
* Recordatorios programados: Todos los días a las 8:00 AM se envían recordatorios de eventos programados para ese día.
* Configuración SMTP: Los datos de correo están en app.py y deben mantenerse actualizados y seguros.

# 8. recomendaciones finales

* No editar archivos JSON manualmente salvo emergencia.
* No eliminar archivos de transacciones sin respaldo.
* Mantener actualizado el correo de los usuarios para asegurar la recepción de notificaciones.
* Consultar la sección de ayuda o contactar al soporte técnico ante cualquier duda.

# 9. contacto y soporte

Para soporte técnico, contactar al responsable de TI de CEVAZ o al desarrollador del sistema.