La estructura del proyecto es así:

* Carpeta Original
  + - Carpeta app
      * Carpeta Controllers
        + Auth\_controller.py
        + User\_controller.py
        + Incidencias\_controller.py
      * Carpeta Models
        + User\_model.py
      * Carpeta Views
        + Carpeta Static

Carpeta CSS

Bienvenida.css

Login.css

Carpeta Img

Agregar Usuario.png

Incidencia\_usuario.png

Incidencia.png

Incidencias\_recibidas.png

Logo-transparente.ico

Mis\_incidencias.png

Ver Usuario.png

* + - * + Carpeta JS

Incidencia.js

* + - * + Carpeta Templates

Agregar\_usuario.html

Bienvenida.html

Editar\_usuario.html

Incidencia.html

Incidencias\_usuario.html

Login.html

Mis\_solicitudes.html

Solicitudes\_recibidas.html

Ver\_incidencia.html

Ver\_registros.html

* + - * \_\_Init\_\_.py
* Run.py

Auth\_controller.py

from **flask** import **Blueprint**, **render\_template**, request, **redirect**, **url\_for**, **flash**, session

from **app**.**models**.**user\_model** import **UserModel**

import **re**

*# Crear un Blueprint en lugar de una app Flask*

auth\_bp = **Blueprint**('auth', \_\_name\_\_, template\_folder="views/templates")

user\_model = **UserModel**()

**@auth\_bp.route**('/')

def **login**():

    return **render\_template**('login.html')

**@auth\_bp.route**('/login', methods=['GET', 'POST'])

def **do\_login**():

    if request.method == 'POST':

        login\_input = request.form['login\_input']

        password = request.form['password']

        resultado = user\_model.**authenticate\_user**(login\_input, password)

        if resultado["success"]:

            usuario = resultado["data"]

            session['numNomina'] = usuario.numNomina

            session['user'] = f"{usuario.nombre} {usuario.apellidoPaterno} {usuario.apellidoMaterno}"

            session['rol'] = usuario.nombreRol

*# Redirigir según el rol*

            if usuario.nombreRol == 'Usuario':

                return **redirect**(**url\_for**('auth.bienvenida'))  *# Página de incidencias (futuro)*

            else:

                return **redirect**(**url\_for**('auth.bienvenida'))  *# Página de bienvenida estándar*

        else:

**flash**("Credenciales incorrectas", "error")

    return **render\_template**('login.html')

**@auth\_bp.route**('/bienvenida')

def **bienvenida**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

    return **render\_template**('bienvenida.html', username=session['user'])

**@auth\_bp.route**('/logout')

def **logout**():

    session.**pop**('user', None)

    return **redirect**(**url\_for**('auth.login'))

*# Función de validación (agrégala al inicio del archivo)*

def **es\_correo\_valido**(correo):

    patron = r"^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$"

    return **re**.**match**(patron, correo) is not None

*# ... (otras rutas y código existente)*

**@auth\_bp.route**("/enviar\_correo", methods=["POST"])

def **enviar\_correo**():

    if "user" not in session:

        return **redirect**(**url\_for**("auth.login"))

*# Obtener y procesar correos*

    destinatarios\_raw = request.form.**get**("destinatario", "")

    destinatarios = [email.**strip**() for email in destinatarios\_raw.**split**(",") if email.**strip**()]

*# Validar correos*

    correos\_invalidos = [email for email in destinatarios if not **es\_correo\_valido**(email)]

    if correos\_invalidos:

**flash**(f"Correos inválidos: {', '.**join**(correos\_invalidos)}", "error")

        return **redirect**(**url\_for**("auth.bienvenida"))

    if not destinatarios:

**flash**("Debes ingresar al menos un correo válido", "error")

        return **redirect**(**url\_for**("auth.bienvenida"))

*# Obtener asunto y cuerpo*

    asunto = request.form.**get**("asunto", "")

    cuerpo = request.form.**get**("cuerpo", "")

*# Enviar correo*

    if user\_model.**enviar\_correo**(destinatarios, asunto, cuerpo):

**flash**(f"Correo enviado exitosamente a {**len**(destinatarios)} destinatarios", "success")

    else:

**flash**("Error al enviar el correo", "error")

    return **redirect**(**url\_for**("auth.bienvenida"))

incidencias\_controller.py

from **flask** import **Blueprint**, **render\_template**, request, **flash**, **redirect**, **url\_for**, session

from **app**.**models**.**user\_model** import **UserModel**

incidencias\_bp = **Blueprint**('incidencias', \_\_name\_\_)

user\_model = **UserModel**()

**@incidencias\_bp.route**('/procesar\_incidencia/<int:idIncidencia>', methods=['POST'])

def **procesar\_incidencia**(idIncidencia):

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

    incidencia = user\_model.**get\_incidencia\_by\_id**(idIncidencia)

    if not incidencia:

**flash**("Incidencia no encontrada", "error")

        return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

    if incidencia['estatus'] in ['Aprobada', 'Rechazada']:

**flash**("Esta incidencia ya ha sido procesada y no puede modificarse", "error")

        return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

    if 'jefe\_directo' not in incidencia:

**flash**("Error en los datos de la incidencia", "error")

        return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

    current\_user\_nomina = session['numNomina']

    accion = request.form.**get**('accion')

    comentarios = request.form.**get**('comentarios', '')

*# Verificar si ya está aprobada/rechazada*

    if incidencia['estatus'] in ['Aprobada', 'Rechazada']:

**flash**("Esta incidencia ya ha sido procesada y no puede modificarse", "warning")

        return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

    if incidencia['estatus'] == 'Pendiente Supervisor':

        if current\_user\_nomina != incidencia['jefe\_directo']:

**flash**("No tienes permiso para procesar esta incidencia", "error")

            return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

        if accion == 'aprobar':

            if not user\_model.**actualizar\_estatus\_incidencia**(

                idIncidencia,

                'Pendiente Gerente',

                current\_user\_nomina,

                comentarios

            ):

**flash**("Error al actualizar el estado", "error")

                return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

*# Cuando el supervisor aprueba y envía al usuario*

            user\_model.**enviar\_notificacion\_incidencia**(

                incidencia['numNomina\_solicitante'],

                'Aprobada',

                incidencia['motivo'],

                incidencia['fecha\_inicio'],

                incidencia['fecha\_fin'],

                aprobador\_rol='supervisor'

            )

*# Cuando el supervisor aprueba y envía al gerente*

            user\_model.**enviar\_notificacion\_incidencia**(

                incidencia['numNomina\_solicitante'],

                'Nueva solicitud para revisión',

                incidencia['motivo'],

                incidencia['fecha\_inicio'],

                incidencia['fecha\_fin'],

                incidencia['gerente\_responsable']

            )

*# Obtener correo del gerente (jefe del supervisor)*

            gerente = user\_model.**get\_user\_by\_numNomina**(incidencia['gerente\_responsable'])

            if not gerente:

**flash**("No se encontró al gerente responsable", "error")

                return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

            solicitante = user\_model.**get\_user\_by\_numNomina**(incidencia['numNomina\_solicitante'])

*# 4. Enviar notificación al gerente*

            if not user\_model.**enviar\_notificacion\_gerente**(

                incidencia\_id=idIncidencia,

                gerente\_nomina=incidencia['gerente\_responsable'],

                solicitante\_nombre=f"{solicitante['nombre']} {solicitante['apellidoPaterno']}",

                supervisor\_nombre=session['user'],

                motivo=incidencia['motivo'],

                fecha\_inicio=incidencia['fecha\_inicio'],

                fecha\_fin=incidencia['fecha\_fin'],

                comentarios=comentarios

            ):

**flash**("Incidencia aprobada pero no se pudo notificar al gerente", "warning")

            else:

**flash**("Incidencia aprobada y gerente notificado", "success")

*# Notificar al EMPLEADO que su solicitud avanzó*

*# Cuando se notifica al usuario*

            user\_model.**enviar\_notificacion\_incidencia**(

                incidencia['numNomina\_solicitante'],

                'Aprobada por supervisor - Pendiente de gerente',

                incidencia['motivo'],

                incidencia['fecha\_inicio'],

                incidencia['fecha\_fin'],

                None,  *# destinatario (None = usuario solicitante)*

                f"Tu solicitud ha sido aprobada por tu supervisor y está pendiente de aprobación por el gerente. Comentarios: {comentarios}"

            )

**flash**("Incidencia aprobada por supervisor y enviada a gerente", "success")

        elif accion == 'rechazar':

*# Cambiar estado directamente a Rechazada (sin pasar por gerente)*

            if not user\_model.**actualizar\_estatus\_incidencia**(

                idIncidencia,

                'Rechazada',

                current\_user\_nomina,

                comentarios

            ):

**flash**("Error al actualizar la incidencia", "error")

                return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

*# Notificar solo al usuario solicitante*

            user\_model.**enviar\_notificacion\_incidencia**(

                incidencia['numNomina\_solicitante'],

                'Rechazada por tu supervisor',

                incidencia['motivo'],

                incidencia['fecha\_inicio'],

                incidencia['fecha\_fin']

            )

**flash**("Incidencia rechazada por supervisor", "success")

    elif incidencia['estatus'] == 'Pendiente Gerente':

*# Solo el gerente puede aprobar en esta etapa*

        if current\_user\_nomina != incidencia['gerente\_responsable']:

**flash**("No tienes permiso para procesar esta incidencia", "error")

            return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

        if accion == 'aprobar':

            user\_model.**actualizar\_estatus\_incidencia**(

                idIncidencia,

                'Aprobada',

                current\_user\_nomina,

                comentarios

            )

*# Cuando se notifica al usuario*

            user\_model.**enviar\_notificacion\_incidencia**(

                incidencia['numNomina\_solicitante'],

                'Aprobada',

                incidencia['motivo'],

                incidencia['fecha\_inicio'],

                incidencia['fecha\_fin'],

                aprobador\_rol='gerente'

            )

**flash**("Incidencia aprobada por gerente", "success")

        elif accion == 'rechazar':

            user\_model.**actualizar\_estatus\_incidencia**(

                idIncidencia,

                'Rechazada',

                current\_user\_nomina,

                comentarios

            )

*# Notificar al usuario*

*# Cuando se notifica al usuario*

            user\_model.**enviar\_notificacion\_incidencia**(

                incidencia['numNomina\_solicitante'],

                'Rechazada',

                incidencia['motivo'],

                incidencia['fecha\_inicio'],

                incidencia['fecha\_fin'],

                aprobador\_rol='gerente'

            )

**flash**("Incidencia rechazada por gerente", "success")

    return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

**@incidencias\_bp.route**('/crear\_incidencia', methods=['GET', 'POST'])

def **crear\_incidencia**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

    if request.method == 'POST':

*# Obtener los datos del formulario*

        numNomina\_solicitante = session['numNomina']

        nombre\_solicitante = request.form.**get**('nombre')

        apellido\_paterno = request.form.**get**('apellido\_paterno')

        apellido\_materno = request.form.**get**('apellido\_materno')

        fecha\_solicitud = request.form.**get**('fecha\_solicitud')

        puesto = request.form.**get**('puesto')

        departamento = request.form.**get**('departamento')

        dias\_vacaciones = request.form.**get**('dias\_vacaciones')

        motivo = request.form.**get**('motivo')

        fecha\_inicio = request.form.**get**('fecha\_inicio')

        fecha\_fin = request.form.**get**('fecha\_fin')

        num\_dias = request.form.**get**('num\_dias')

        observaciones = request.form.**get**('comentarios')

*# Obtener el jefe directo del usuario*

        jefe\_directo = user\_model.**get\_jefe\_directo**(numNomina\_solicitante)

*# Guardar la incidencia en la base de datos*

        idIncidencia = user\_model.**crear\_incidencia**(

            numNomina\_solicitante,

            nombre\_solicitante,

            apellido\_paterno,

            apellido\_materno,

            fecha\_solicitud,

            puesto,

            departamento,

            dias\_vacaciones,

            motivo,

            fecha\_inicio,

            fecha\_fin,

            num\_dias,

            observaciones,

            jefe\_directo

        )

        if idIncidencia:

            return **redirect**(**url\_for**('incidencias.mis\_solicitudes'))

        else:

**flash**("Error al enviar la solicitud", "error")

    return **render\_template**('incidencia.html')

**@incidencias\_bp.route**('/ver\_incidencia/<int:idIncidencia>/<origen>', methods=['GET'])

def **ver\_incidencia**(idIncidencia, origen):

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

*# Obtener la incidencia*

    incidencia = user\_model.**get\_incidencia\_by\_id**(idIncidencia)

    if not incidencia:

**flash**("Incidencia no encontrada", "error")

        return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

    if request.method == 'POST':

        accion = request.form.**get**('accion')  *# 'aprobar' o 'rechazar'*

        if accion == 'aprobar':

            user\_model.**actualizar\_estatus\_incidencia**(idIncidencia, 'Aprobada')

*# Enviar correo al usuario*

            user\_model.enviar\_correo\_incidencia(incidencia.numNomina\_solicitante, 'Aprobada')

        elif accion == 'rechazar':

            user\_model.**actualizar\_estatus\_incidencia**(idIncidencia, 'Rechazada')

*# Enviar correo al usuario*

            user\_model.enviar\_correo\_incidencia(incidencia.numNomina\_solicitante, 'Rechazada')

        return **redirect**(**url\_for**('incidencias.solicitudes\_recibidas'))

    return **render\_template**('ver\_incidencia.html', incidencia=incidencia, origen=origen)

**@incidencias\_bp.route**('/solicitudes\_recibidas')

def **solicitudes\_recibidas**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

*# Obtener el parámetro de orden (ascendente o descendente)*

    orden = request.args.**get**('orden', 'asc')  *# Por defecto, orden ascendente*

*# Obtener las solicitudes recibidas por el jefe directo*

    numNomina\_jefe = session['numNomina']

    solicitudes = user\_model.**get\_solicitudes\_recibidas**(numNomina\_jefe, orden)

    return **render\_template**('solicitudes\_recibidas.html', solicitudes=solicitudes, orden=orden)

**@incidencias\_bp.route**('/mis\_solicitudes')

def **mis\_solicitudes**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

*# Obtener el parámetro de orden (ascendente o descendente)*

    orden = request.args.**get**('orden', 'asc')  *# Por defecto, orden ascendente*

*# Obtener las solicitudes enviadas por el usuario actual*

    numNomina = session['numNomina']

    solicitudes = user\_model.**get\_solicitudes\_enviadas**(numNomina, orden)

    return **render\_template**('mis\_solicitudes.html', solicitudes=solicitudes, orden=orden)

**@incidencias\_bp.route**('/ver\_incidencias')

def **ver\_incidencias**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

*# Obtener puestos y departamentos desde la base de datos*

    puestos = user\_model.**get\_puestos**()

    departamentos = user\_model.**get\_departamentos**()

*# Obtener los días de vacaciones restantes del usuario actual*

    vacaciones = user\_model.**get\_vacaciones**(session['numNomina'])

    return **render\_template**('incidencia.html', puestos=puestos, departamentos=departamentos, vacaciones=vacaciones)

**@incidencias\_bp.route**('/crear\_incidencia\_usuario/<int:numNomina>/<origen>', methods=['GET', 'POST'])

def **crear\_incidencia\_usuario**(numNomina, origen):

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

*# Obtener los datos del usuario*

    usuario = user\_model.**get\_user\_by\_numNomina**(numNomina)

    if not usuario:

**flash**("Usuario no encontrado", "error")

        return **redirect**(**url\_for**('auth.bienvenida'))

*# Obtener la fecha actual*

    from **datetime** import **datetime**

    fecha\_solicitud = **datetime**.**now**().**strftime**('%Y-%m-%d')

*# Obtener el nombre del departamento y puesto del usuario*

    departamento\_usuario = user\_model.**get\_departamento\_by\_id**(usuario['idDepartamento'])

    puesto\_usuario = user\_model.**get\_puesto\_by\_id**(usuario['idPuesto'])

*# Obtener todos los puestos y departamentos para los selectores*

    puestos = user\_model.**get\_puestos**()

    departamentos = user\_model.**get\_departamentos**()

    return **render\_template**('incidencia.html',

                           nombre=usuario['nombre'],

                           apellido\_paterno=usuario['apellidoPaterno'],

                           apellido\_materno=usuario['apellidoMaterno'],

                           fecha\_solicitud=fecha\_solicitud,

                           puesto=usuario['idPuesto'],

                           nombre\_puesto=puesto\_usuario['nombrePuesto'],  *# Acceder al valor usando la clave*

                           no\_nomina=usuario['numNomina'],

                           departamento=usuario['idDepartamento'],

                           nombre\_departamento=departamento\_usuario['nombreDepartamento'],  *# Acceder al valor usando la clave*

                           dias\_vacaciones=usuario['diasVacaciones'],  *# Días de vacaciones*

                           puestos=puestos,

                           departamentos=departamentos,

                           origen=origen)  *# Pasamos el origen a la plantilla*

user\_controller.py

from **flask** import **Blueprint**, **render\_template**, request, **redirect**, **url\_for**, **flash**, session

from **app**.**models**.**user\_model** import **UserModel**

from **flask** import **jsonify**

*# Crear un Blueprint para las rutas de usuario*

user\_bp = **Blueprint**('user', \_\_name\_\_)

user\_model = **UserModel**()

*# Ruta para agregar usuario*

**@user\_bp.route**('/agregar\_usuario', methods=['GET', 'POST'])

def **agregar\_usuario**():

    if 'rol' not in session or session['rol'] != 'Admin':

**flash**("No tienes permiso para acceder a esta página", "error")

        return **redirect**(**url\_for**('auth.bienvenida'))

    mensaje = None

    departamentos = user\_model.**get\_departamentos**()

    roles = user\_model.**get\_roles**()

    usuarios = user\_model.**get\_all\_users**()

    if request.method == 'POST':

        numNomina = request.form.**get**('numNomina')

        nombre = request.form.**get**('nombre')

        apellido\_paterno = request.form.**get**('apellidoPaterno', '')

        apellido\_materno = request.form.**get**('apellidoMaterno', '')

        username = request.form.**get**('username')

        password = request.form.**get**('password')

        idRol = request.form.**get**('idRol')

        idDepartamento = request.form.**get**('idDepartamento')

        idPuesto = request.form.**get**('idPuesto')

        diasVacaciones = request.form.**get**('diasVacaciones', 0)

        correo\_electronico = request.form.**get**('correo\_electronico')

        jefe\_directo = request.form.**get**('jefe\_directo')

        if numNomina and nombre and username and password and idRol and idDepartamento and idPuesto:

            if user\_model.**add\_user**(numNomina, nombre, apellido\_paterno, apellido\_materno, username, password, idRol, idDepartamento, idPuesto, diasVacaciones, correo\_electronico, jefe\_directo):

**flash**("Usuario agregado exitosamente", "success")

                return **redirect**(**url\_for**('user.agregar\_usuario'))  *# Redirigir a la misma página*

            else:

**flash**("Error al agregar usuario", "error")

        else:

**flash**("Por favor, completa todos los campos obligatorios", "error")

    return **render\_template**('agregar\_usuario.html', mensaje=mensaje, departamentos=departamentos, roles=roles, usuarios=usuarios)

**@user\_bp.route**('/ver\_registros', methods=['GET', 'POST'])

def **ver\_registros**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

    busqueda = request.form.**get**('numNomina') if request.method == 'POST' else None

*# Obtener todos los usuarios con detalles (incluyendo rol, correo y jefe)*

    registros = user\_model.**get\_all\_users\_with\_details**()

*# Filtrar por número de nómina si se proporciona una búsqueda*

    if busqueda:

        usuario = user\_model.**get\_user\_by\_numNomina**(busqueda)

        registros = [usuario] if usuario else []

    return **render\_template**('ver\_registros.html', registros=registros, busqueda=busqueda, username=session['user'])

**@user\_bp.route**('/borrar\_usuario/<int:numNomina>', methods=['POST'])

def **borrar\_usuario**(numNomina):

    if 'rol' not in session or session['rol'] != 'Admin':

**flash**("No tienes permiso para realizar esta acción", "error")

        return **redirect**(**url\_for**('auth.bienvenida'))

    if user\_model.**delete\_user**(numNomina):

**flash**("Usuario borrado exitosamente", "success")

    else:

**flash**("Error al borrar usuario", "error")

    return **redirect**(**url\_for**('auth.bienvenida'))

**@user\_bp.route**('/editar\_usuario/<int:numNomina>', methods=['GET', 'POST'])

def **editar\_usuario**(numNomina):

    if 'rol' not in session or session['rol'] not in ['Admin', 'Supervisor', 'Gerente']:

**flash**("No tienes permiso para acceder a esta página", "error")

        return **redirect**(**url\_for**('auth.bienvenida'))

    usuario = user\_model.**get\_user\_by\_numNomina**(numNomina)

    departamentos = user\_model.**get\_departamentos**()

    puestos = user\_model.**get\_puestos**()

    roles = user\_model.**get\_roles**()

    usuarios = user\_model.**get\_all\_users**()  *# Obtener todos los usuarios para seleccionar jefe directo*

    if not usuario:

**flash**("Usuario no encontrado", "error")

        return **redirect**(**url\_for**('auth.bienvenida'))

    if request.method == 'POST':

        nombre = request.form['nombre']

        apellido\_paterno = request.form['apellidoPaterno']

        apellido\_materno = request.form['apellidoMaterno']

        username = request.form['username']

        password = request.form['password']

        idDepartamento = request.form['idDepartamento']

        idPuesto = request.form['idPuesto']

        idRol = request.form['idRol']

        diasVacaciones = request.form['diasVacaciones']

        correo\_electronico = request.form['correo\_electronico']  *# Nuevo campo*

        jefe\_directo = request.form['jefe\_directo']  *# Nuevo campo*

        if user\_model.**update\_user**(numNomina, nombre, apellido\_paterno, apellido\_materno, username, password, idDepartamento, idPuesto, idRol, diasVacaciones, correo\_electronico, jefe\_directo):

**flash**("Usuario actualizado exitosamente", "success")

            return **redirect**(**url\_for**('user.ver\_registros'))

        else:

**flash**("Error al actualizar usuario", "error")

    return **render\_template**('editar\_usuario.html', usuario=usuario, departamentos=departamentos, puestos=puestos, roles=roles, usuarios=usuarios)

**@user\_bp.route**('/obtener\_puestos\_por\_departamento/<int:idDepartamento>', methods=['GET'])

def **obtener\_puestos\_por\_departamento**(idDepartamento):

    puestos = user\_model.**get\_puestos\_por\_departamento**(idDepartamento)

    return **jsonify**(puestos)

**@user\_bp.route**('/incidencias\_usuario')

def **incidencias\_usuario**():

    if 'user' not in session:

        return **redirect**(**url\_for**('auth.login'))

*# Obtener todos los usuarios con detalles*

    registros = user\_model.**get\_all\_users\_with\_details**()

    return **render\_template**('incidencias\_usuario.html', registros=registros, username=session['user'])

user\_model.py

import **pyodbc**

import **smtplib**

from **email**.**mime**.**text** import **MIMEText**

from **flask** import current\_app

class **UserModel**:

    def **\_\_init\_\_**(self):

        self.connection\_string = (

            "DRIVER={ODBC Driver 17 for SQL Server};"

            "SERVER=localhost;"

            "DATABASE=Prueba\_10;"

            "UID=sa;"

            "PWD=root"

        )

    def **get\_connection**(self):

        return **pyodbc**.**connect**(self.connection\_string)

    def **authenticate\_user**(self, login\_input, password):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Primero verificar si existe el usuario/nómina*

            if login\_input.isdigit():

                query\_user = "SELECT numNomina FROM usuarios WHERE numNomina = ?"

                query\_credenciales = "SELECT password FROM credenciales WHERE numNomina = ?"

            else:

                query\_user = "SELECT numNomina FROM credenciales WHERE username = ?"

                query\_credenciales = "SELECT password FROM credenciales WHERE username = ?"

*# 1. Verificar existencia del usuario*

            cursor.**execute**(query\_user, (login\_input,))

            usuario\_existente = cursor.**fetchone**()

            if not usuario\_existente:

                return {"success": False, "error": "user\_not\_found"}

*# 2. Verificar contraseña*

            cursor.**execute**(query\_credenciales, (login\_input,))

            credencial = cursor.**fetchone**()

            if credencial.password != password:

                return {"success": False, "error": "wrong\_password"}

*# 3. Obtener datos completos del usuario*

            cursor.**execute**("""

                SELECT u.numNomina, u.nombre, u.apellidoPaterno, u.apellidoMaterno,

                    c.username, r.nombreRol

                FROM usuarios u

                INNER JOIN credenciales c ON u.numNomina = c.numNomina

                INNER JOIN usuario\_rol ur ON u.numNomina = ur.numNomina

                INNER JOIN roles r ON ur.idRol = r.idRol

                WHERE c.username = ? OR u.numNomina = ?

            """, (login\_input, login\_input))

            usuario = cursor.**fetchone**()

            return {"success": True, "data": usuario}

        except **Exception** as e:

**print**(f"Error en autenticación: {e}")

            return {"success": False, "error": "server\_error"}

        finally:

            cursor.**close**()

            conn.**close**()

    def **add\_user**(self, numNomina, nombre, apellido\_paterno, apellido\_materno,

                username, password, idRol, idDepartamento, idPuesto,

                diasVacaciones, correo\_electronico, jefe\_directo):

        """Agrega un nuevo usuario con todos los campos requeridos"""

        try:

            conn = self.**get\_connection**()

            cursor = conn.**cursor**()

*# Validar campos requeridos*

            if not **all**([numNomina, nombre, username, password, idRol,

                    idDepartamento, idPuesto, correo\_electronico]):

                raise **ValueError**("Todos los campos obligatorios deben ser proporcionados")

*# Insertar usuario*

            cursor.**execute**("""

                INSERT INTO usuarios (

                    numNomina, nombre, apellidoPaterno, apellidoMaterno,

                    idDepartamento, idPuesto, diasVacaciones,

                    correo\_electronico, jefe\_directo

                ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)

            """, (numNomina, nombre, apellido\_paterno, apellido\_materno,

                idDepartamento, idPuesto, diasVacaciones,

                correo\_electronico, jefe\_directo))

*# Insertar credenciales*

            cursor.**execute**("""

                INSERT INTO credenciales (numNomina, username, password)

                VALUES (?, ?, ?)

            """, (numNomina, username, password))

*# Asignar rol*

            cursor.**execute**("""

                INSERT INTO usuario\_rol (numNomina, idRol)

                VALUES (?, ?)

            """, (numNomina, idRol))

            conn.**commit**()

            return True

        except **Exception** as e:

**print**(f"Error al agregar usuario: {e}")

            conn.**rollback**()

            return False

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_all\_users**(self):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        cursor.**execute**("""

            SELECT u.numNomina, u.nombre, u.apellidoPaterno, u.apellidoMaterno, c.username

            FROM usuarios u

            INNER JOIN credenciales c ON u.numNomina = c.numNomina

        """)

        registros = cursor.**fetchall**()

        cursor.**close**()

        conn.**close**()

        return registros

    def **update\_user**(self, numNomina, nombre, apellido\_paterno, apellido\_materno, username, password, idDepartamento, idPuesto, idRol, diasVacaciones, correo\_electronico, jefe\_directo):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Actualizar tabla usuarios*

            cursor.**execute**("""

                UPDATE usuarios

                SET nombre = ?, apellidoPaterno = ?, apellidoMaterno = ?, idDepartamento = ?, idPuesto = ?, diasVacaciones = ?, correo\_electronico = ?, jefe\_directo = ?

                WHERE numNomina = ?

            """, (nombre, apellido\_paterno, apellido\_materno, idDepartamento, idPuesto, diasVacaciones, correo\_electronico, jefe\_directo, numNomina))

*# Actualizar tabla credenciales*

            cursor.**execute**("""

                UPDATE credenciales

                SET username = ?

                WHERE numNomina = ?

            """, (username, numNomina))

*# Actualizar tabla credenciales*

            cursor.**execute**("""

                UPDATE credenciales

                SET password = ?

                WHERE numNomina = ?

            """, (password, numNomina))

*# Actualizar tabla usuario\_rol*

            cursor.**execute**("""

                UPDATE usuario\_rol

                SET idRol = ?

                WHERE numNomina = ?

            """, (idRol, numNomina))

            conn.**commit**()

            return True

        except **Exception** as e:

**print**(f"Error al actualizar usuario: {e}")

            return False

        finally:

            cursor.**close**()

            conn.**close**()

    def **delete\_user**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Eliminar de usuario\_rol*

            cursor.**execute**("DELETE FROM usuario\_rol WHERE numNomina = ?", (numNomina,))

*# Eliminar de credenciales*

            cursor.**execute**("DELETE FROM credenciales WHERE numNomina = ?", (numNomina,))

*# Eliminar de usuarios*

            cursor.**execute**("DELETE FROM usuarios WHERE numNomina = ?", (numNomina,))

            conn.**commit**()

            return True

        except **Exception** as e:

**print**(f"Error al borrar usuario: {e}")

            return False

        finally:

            cursor.**close**()

            conn.**close**()

    def **enviar\_correo**(self, destinatario, asunto, cuerpo):

        try:

*# Configurar el mensaje*

            mensaje = **MIMEText**(cuerpo)

            mensaje["Subject"] = asunto

            mensaje["From"] = current\_app.config["MAIL\_USERNAME"]

            mensaje["To"] = destinatario

*# Conectar al servidor SMTP*

            with **smtplib**.**SMTP**(

                current\_app.config["MAIL\_SERVER"],

                current\_app.config["MAIL\_PORT"]

            ) as server:

                server.**starttls**()

                server.**login**(

                    current\_app.config["MAIL\_USERNAME"],

                    current\_app.config["MAIL\_PASSWORD"]

                )

                server.**send\_message**(mensaje)

            return True

        except **Exception** as e:

**print**(f"Error al enviar correo: {e}")

            return False

    def **get\_roles**(self):

            conn = self.**get\_connection**()

            cursor = conn.**cursor**()

            try:

                cursor.**execute**("SELECT idRol, nombreRol FROM roles")

                roles = cursor.**fetchall**()

                return roles

            except **Exception** as e:

**print**(f"Error al obtener roles: {e}")

                return []

            finally:

                cursor.**close**()

                conn.**close**()

    def **get\_user\_by\_numNomina**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT

                    u.numNomina,

                    u.nombre,

                    u.apellidoPaterno,

                    u.apellidoMaterno,

                    c.username,

                    c.password,

                    u.idDepartamento,

                    u.idPuesto,

                    u.diasVacaciones,

                    ur.idRol,

                    u.correo\_electronico,

                    u.jefe\_directo

                FROM usuarios u

                INNER JOIN credenciales c ON u.numNomina = c.numNomina

                INNER JOIN usuario\_rol ur ON u.numNomina = ur.numNomina

                WHERE u.numNomina = ?

            """, (numNomina,))

            usuario = cursor.**fetchone**()

            if usuario:

*# Convertir el resultado en un diccionario*

                usuario\_dict = {

                    'numNomina': usuario.numNomina,

                    'nombre': usuario.nombre,

                    'apellidoPaterno': usuario.apellidoPaterno,

                    'apellidoMaterno': usuario.apellidoMaterno,

                    'username': usuario.username,

                    'password': usuario.password,

                    'idDepartamento': usuario.idDepartamento,

                    'idPuesto': usuario.idPuesto,

                    'diasVacaciones': usuario.diasVacaciones,

                    'idRol': usuario.idRol,

                    'correo\_electronico': usuario.correo\_electronico,

                    'jefe\_directo': usuario.jefe\_directo,

                }

                return usuario\_dict

            return None

        except **Exception** as e:

**print**(f"Error al obtener usuario: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_vacaciones**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        cursor.**execute**("SELECT diasVacaciones FROM vacaciones WHERE numNomina = ?", (numNomina,))

        vacaciones = cursor.**fetchone**()

        cursor.**close**()

        conn.**close**()

        return vacaciones[0] if vacaciones else 0

    def **get\_departamentos**(self):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        cursor.**execute**("SELECT idDepartamento, nombreDepartamento FROM departamentos")

        departamentos = cursor.**fetchall**()

        cursor.**close**()

        conn.**close**()

        return departamentos

    def **get\_puestos**(self):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        cursor.**execute**("SELECT idPuesto, nombrePuesto FROM puestos")

        puestos = cursor.**fetchall**()

        cursor.**close**()

        conn.**close**()

        return puestos

    def **get\_puestos\_por\_departamento**(self, idDepartamento):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Obtener los puestos asociados al departamento*

            cursor.**execute**("""

                SELECT p.idPuesto, p.nombrePuesto

                FROM puestos p

                INNER JOIN departamento\_puesto dp ON p.idPuesto = dp.idPuesto

                WHERE dp.idDepartamento = ?

            """, (idDepartamento,))

            puestos = cursor.**fetchall**()

            return [{"idPuesto": puesto.idPuesto, "nombrePuesto": puesto.nombrePuesto} for puesto in puestos]

        except **Exception** as e:

**print**(f"Error al obtener puestos por departamento: {e}")

            return []

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_departamento\_by\_id**(self, idDepartamento):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("SELECT nombreDepartamento FROM departamentos WHERE idDepartamento = ?", (idDepartamento,))

            departamento = cursor.**fetchone**()

            if departamento:

*# Convertir la fila en un diccionario*

                return {'nombreDepartamento': departamento[0]}  *# Acceder al primer campo usando índice 0*

            return None

        except **Exception** as e:

**print**(f"Error al obtener departamento: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_puesto\_by\_id**(self, idPuesto):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("SELECT nombrePuesto FROM puestos WHERE idPuesto = ?", (idPuesto,))

            puesto = cursor.**fetchone**()

            if puesto:

*# Convertir la fila en un diccionario*

                return {'nombrePuesto': puesto[0]}  *# Acceder al primer campo usando índice 0*

            return None

        except **Exception** as e:

**print**(f"Error al obtener puesto: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_all\_users\_with\_details**(self):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT

                    u.numNomina,

                    u.nombre,

                    u.apellidoPaterno,

                    u.apellidoMaterno,

                    c.username,

                    d.nombreDepartamento,

                    p.nombrePuesto,

                    u.diasVacaciones,

                    r.nombreRol,               -- Nuevo campo: rol

                    u.correo\_electronico,       -- Nuevo campo: correo electrónico

                    j.nombre AS nombre\_jefe,    -- Nuevo campo: nombre del jefe directo

                    j.apellidoPaterno AS apellido\_jefe

                FROM usuarios u

                INNER JOIN credenciales c ON u.numNomina = c.numNomina

                LEFT JOIN departamentos d ON u.idDepartamento = d.idDepartamento

                LEFT JOIN puestos p ON u.idPuesto = p.idPuesto

                LEFT JOIN usuario\_rol ur ON u.numNomina = ur.numNomina

                LEFT JOIN roles r ON ur.idRol = r.idRol

                LEFT JOIN usuarios j ON u.jefe\_directo = j.numNomina  -- Join para obtener el jefe directo

            """)

            registros = cursor.**fetchall**()

            return registros

        except **Exception** as e:

**print**(f"Error al obtener usuarios con detalles: {e}")

            return []

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_solicitudes\_enviadas**(self, numNomina, orden='asc'):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            query = """

                SELECT

                    idIncidencia,

                    numNomina\_solicitante,

                    fecha\_solicitud,

                    motivo,

                    estatus

                FROM incidencias

                WHERE numNomina\_solicitante = ?

                ORDER BY idIncidencia {}  -- Ordenar por ID

            """.**format**('ASC' if orden == 'asc' else 'DESC')

            cursor.**execute**(query, (numNomina,))

            return cursor.**fetchall**()

        except **Exception** as e:

**print**(f"Error al obtener solicitudes enviadas: {e}")

            return []

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_solicitudes\_recibidas**(self, numNomina\_jefe, rol\_usuario, orden='asc'):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            query = """

                SELECT idIncidencia, numNomina\_solicitante, fecha\_solicitud,

                    motivo, estatus, aprobado\_por\_supervisor, aprobado\_por\_gerente

                FROM incidencias

                WHERE (jefe\_directo = ? AND (estatus IN ('Pendiente Supervisor', 'Aprobada', 'Rechazada')))

                OR (gerente\_responsable = ? AND (estatus IN ('Pendiente Gerente', 'Aprobada', 'Rechazada')))

                ORDER BY

                    CASE WHEN estatus IN ('Pendiente Supervisor', 'Pendiente Gerente') THEN 1 ELSE 2 END,

                    idIncidencia {}

            """.**format**('ASC' if orden == 'asc' else 'DESC')

            cursor.**execute**(query, (numNomina\_jefe, numNomina\_jefe))

            return cursor.**fetchall**()

        except **Exception** as e:

**print**(f"Error al obtener solicitudes recibidas: {e}")

            return []

        finally:

            cursor.**close**()

            conn.**close**()

    def **enviar\_correo**(self, destinatario, asunto, cuerpo):

        try:

*# Configurar el mensaje*

            mensaje = **MIMEText**(cuerpo)

            mensaje["Subject"] = asunto

            mensaje["From"] = current\_app.config["MAIL\_USERNAME"]

            mensaje["To"] = destinatario

*# Conectar al servidor SMTP*

            with **smtplib**.**SMTP**(

                current\_app.config["MAIL\_SERVER"],

                current\_app.config["MAIL\_PORT"]

            ) as server:

                server.**starttls**()

                server.**login**(

                    current\_app.config["MAIL\_USERNAME"],

                    current\_app.config["MAIL\_PASSWORD"]

                )

                server.**send\_message**(mensaje)

            return True

        except **Exception** as e:

**print**(f"Error al enviar correo: {e}")

            return False

    def **crear\_incidencia**(self, numNomina\_solicitante, nombre\_solicitante, apellido\_paterno,

                    apellido\_materno, fecha\_solicitud, puesto, departamento,

                    dias\_vacaciones, motivo, fecha\_inicio, fecha\_fin, num\_dias,

                    observaciones, jefe\_directo):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Obtener el gerente responsable*

            cursor.**execute**("SELECT jefe\_directo FROM usuarios WHERE numNomina = ?", (jefe\_directo,))

            gerente\_responsable\_result = cursor.**fetchone**()

            gerente\_responsable = gerente\_responsable\_result[0] if gerente\_responsable\_result else jefe\_directo

*# Insertar la incidencia*

            cursor.**execute**("""

                INSERT INTO incidencias (

                    numNomina\_solicitante, nombre\_solicitante, apellido\_paterno, apellido\_materno,

                    fecha\_solicitud, puesto, departamento, dias\_vacaciones, motivo,

                    fecha\_inicio, fecha\_fin, num\_dias, observaciones, jefe\_directo, gerente\_responsable

                ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)

            """, (

                numNomina\_solicitante, nombre\_solicitante, apellido\_paterno, apellido\_materno,

                fecha\_solicitud, puesto, departamento, dias\_vacaciones, motivo,

                fecha\_inicio, fecha\_fin, num\_dias, observaciones, jefe\_directo, gerente\_responsable

            ))

            conn.**commit**()

*# Notificar al supervisor*

            self.**enviar\_notificacion\_incidencia**(

                numNomina\_solicitante,

                'Nueva solicitud creada',

                motivo,

                fecha\_inicio,

                fecha\_fin,

                jefe\_directo,  *# destinatario (supervisor)*

                f"Nueva solicitud creada por {nombre\_solicitante} {apellido\_paterno}"

            )

            return True

        except **Exception** as e:

**print**(f"Error al crear incidencia: {e}")

            conn.**rollback**()

            return False

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_jefe\_directo**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT jefe\_directo FROM usuarios

                WHERE numNomina = ?

            """, (numNomina,))

            jefe\_directo = cursor.**fetchone**()

            return jefe\_directo[0] if jefe\_directo else None

        except **Exception** as e:

**print**(f"Error al obtener jefe directo: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_correo\_jefe\_directo**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Obtener el número de nómina del jefe directo*

            cursor.**execute**("""

                SELECT jefe\_directo FROM usuarios

                WHERE numNomina = ?

            """, (numNomina,))

            jefe\_directo = cursor.**fetchone**()

            if jefe\_directo:

*# Obtener el correo electrónico del jefe directo*

                cursor.**execute**("""

                    SELECT correo\_electronico FROM usuarios

                    WHERE numNomina = ?

                """, (jefe\_directo[0],))

                correo\_jefe = cursor.**fetchone**()

                return correo\_jefe[0] if correo\_jefe else None

            return None

        except **Exception** as e:

**print**(f"Error al obtener correo del jefe directo: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_incidencia\_by\_id**(self, idIncidencia):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT i.\*,

                    s.nombre as nombre\_supervisor,

                    s.apellidoPaterno as apellido\_supervisor,

                    g.nombre as nombre\_gerente,

                    g.apellidoPaterno as apellido\_gerente

                FROM incidencias i

                LEFT JOIN usuarios s ON i.aprobado\_por\_supervisor = s.numNomina

                LEFT JOIN usuarios g ON i.aprobado\_por\_gerente = g.numNomina

                WHERE i.idIncidencia = ?

            """, (idIncidencia,))

            incidencia = cursor.**fetchone**()

            if incidencia:

*# Convertir el resultado en un diccionario para acceder a los campos por nombre*

                incidencia\_dict = {

                    'idIncidencia': incidencia.idIncidencia,

                    'numNomina\_solicitante': incidencia.numNomina\_solicitante,

                    'nombre\_solicitante': incidencia.nombre\_solicitante,

                    'apellido\_paterno': incidencia.apellido\_paterno,

                    'apellido\_materno': incidencia.apellido\_materno,

                    'fecha\_solicitud': incidencia.fecha\_solicitud,

                    'puesto': incidencia.puesto,

                    'departamento': incidencia.departamento,

                    'dias\_vacaciones': incidencia.dias\_vacaciones,

                    'motivo': incidencia.motivo,

                    'fecha\_inicio': incidencia.fecha\_inicio,

                    'fecha\_fin': incidencia.fecha\_fin,

                    'num\_dias': incidencia.num\_dias,

                    'observaciones': incidencia.observaciones,

                    'estatus': incidencia.estatus,

                    'jefe\_directo' : incidencia.jefe\_directo,

                    'gerente\_responsable' : incidencia.gerente\_responsable,

                    'aprobado\_por\_supervisor' : incidencia.aprobado\_por\_supervisor,

                    'fecha\_aprobacion\_supervisor' : incidencia.fecha\_aprobacion\_supervisor,

                    'aprobado\_por\_gerente' : incidencia.aprobado\_por\_gerente,

                    'fecha\_aprobacion\_gerente' : incidencia.fecha\_aprobacion\_gerente,

                    'comentarios\_supervisor' : incidencia.comentarios\_supervisor,

                    'comentarios\_gerente' : incidencia.comentarios\_gerente

                }

                return incidencia\_dict

            return None

        except **Exception** as e:

**print**(f"Error al obtener incidencia por ID: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **actualizar\_estatus\_incidencia**(self, idIncidencia, estatus, numNomina\_aprobador, comentarios=None):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

*# Si es rechazo, marcamos directamente como Rechazada*

            if estatus == 'Rechazada':

                cursor.**execute**("""

                    UPDATE incidencias

                    SET estatus = 'Rechazada',

                        aprobado\_por\_supervisor = ?,

                        fecha\_aprobacion\_supervisor = GETDATE(),

                        comentarios\_supervisor = ?

                    WHERE idIncidencia = ?

                """, (numNomina\_aprobador, comentarios, idIncidencia))

            else:

*# Lógica normal para otros estados*

                current\_status = self.**get\_estatus\_incidencia**(idIncidencia)

                if current\_status == 'Pendiente Supervisor':

                    cursor.**execute**("""

                        UPDATE incidencias

                        SET estatus = ?,

                            aprobado\_por\_supervisor = ?,

                            fecha\_aprobacion\_supervisor = GETDATE(),

                            comentarios\_supervisor = ?

                        WHERE idIncidencia = ?

                    """, (estatus, numNomina\_aprobador, comentarios, idIncidencia))

                elif current\_status == 'Pendiente Gerente':

                    cursor.**execute**("""

                        UPDATE incidencias

                        SET estatus = ?,

                            aprobado\_por\_gerente = ?,

                            fecha\_aprobacion\_gerente = GETDATE(),

                            comentarios\_gerente = ?

                        WHERE idIncidencia = ?

                    """, (estatus, numNomina\_aprobador, comentarios, idIncidencia))

            conn.**commit**()

            return True

        except **Exception** as e:

**print**(f"Error al actualizar estatus de la incidencia: {e}")

            conn.**rollback**()

            return False

        finally:

            cursor.**close**()

            conn.**close**()

    def **enviar\_notificacion\_incidencia**(self, numNomina, estado, motivo, fecha\_inicio, fecha\_fin, destinatario=None, aprobador\_rol=None):

        """Envía notificación por correo con asunto dinámico"""

        try:

*# Obtener información del solicitante*

            solicitante = self.**get\_user\_by\_numNomina**(numNomina)

            nombre\_solicitante = f"{solicitante['nombre']} {solicitante['apellidoPaterno']}" if solicitante else "El usuario"

*# Determinar el asunto según el estado y el rol del aprobador*

            if estado.lower() == 'aprobada':

                if aprobador\_rol == 'supervisor':

                    asunto = f"Solicitud aprobada por tu supervisor - Pendiente de gerente"

                elif aprobador\_rol == 'gerente':

                    asunto = f"¡Felicidades! Tu solicitud ha sido aprobada por el gerente"

                else:

                    asunto = f"Tu solicitud ha sido aprobada"

            elif estado.lower() == 'rechazada':

                if aprobador\_rol == 'supervisor':

                    asunto = f"Solicitud rechazada por tu supervisor"

                elif aprobador\_rol == 'gerente':

                    asunto = f"Solicitud rechazada por el gerente"

                else:

                    asunto = f"Tu solicitud ha sido rechazada"

            else:

                asunto = f"Actualización en tu solicitud: {estado}"

*# Determinar el destinatario*

            if destinatario is None:

*# Notificar al usuario solicitante*

                correo\_destino = self.**get\_correo\_usuario**(numNomina)

                saludo = f"Hola {nombre\_solicitante},"

            else:

*# Notificar a supervisor/gerente*

                correo\_destino = self.**get\_correo\_usuario**(destinatario)

                aprobador = self.**get\_user\_by\_numNomina**(destinatario)

                nombre\_aprobador = f"{aprobador['nombre']} {aprobador['apellidoPaterno']}" if aprobador else ""

                saludo = f"Hola {nombre\_aprobador}," if nombre\_aprobador else "Hola,"

*# Verificar que tenemos un correo destino*

            if not correo\_destino:

**print**(f"No se puede enviar notificación: no hay correo registrado")

                return False

*# Crear el cuerpo del mensaje*

            cuerpo = f"""

            <html>

            <body>

                <p>{saludo}</p>

                <h3>{asunto}</h3>

                <p><strong>Detalles de la solicitud:</strong></p>

                <ul>

                    <li><strong>Motivo:</strong> {motivo}</li>

                    <li><strong>Fecha de inicio:</strong> {fecha\_inicio}</li>

                    <li><strong>Fecha de fin:</strong> {fecha\_fin}</li>

                </ul>

                <br>

                <p>Por favor inicia sesión en el sistema para más detalles.</p>

                <p>Este es un mensaje automático, no responda a este correo.</p>

            </body>

            </html>

            """

*# Configurar y enviar el mensaje*

            mensaje = **MIMEText**(cuerpo, 'html')

            mensaje['Subject'] = asunto

            mensaje['From'] = current\_app.config["MAIL\_USERNAME"]

            mensaje['To'] = correo\_destino

            with **smtplib**.**SMTP**(

                current\_app.config["MAIL\_SERVER"],

                current\_app.config["MAIL\_PORT"]

            ) as server:

                server.**starttls**()

                server.**login**(

                    current\_app.config["MAIL\_USERNAME"],

                    current\_app.config["MAIL\_PASSWORD"]

                )

                server.**send\_message**(mensaje)

**print**(f"Notificación enviada a {correo\_destino}: {asunto}")

            return True

        except **Exception** as e:

**print**(f"Error al enviar notificación de incidencia: {e}")

            return False

    def **enviar\_notificacion\_gerente**(self, incidencia\_id, gerente\_nomina, solicitante\_nombre,

                              supervisor\_nombre, motivo, fecha\_inicio, fecha\_fin, comentarios):

        try:

*# Obtener correo del gerente*

            correo\_gerente = self.**get\_correo\_usuario**(gerente\_nomina)

            if not correo\_gerente:

**print**(f"No se encontró correo para el gerente {gerente\_nomina}")

                return False

*# Crear enlace directo a la incidencia (ajusta tu URL base)*

            enlace\_incidencia = f"http://tudominio.com/ver\_incidencia/{incidencia\_id}/gerente"

*# Construir el mensaje*

            asunto = f"[URGENTE] Solicitud pendiente de aprobación: {solicitante\_nombre}"

            cuerpo = f"""

            <html>

            <body>

                <h2>Solicitud pendiente de tu aprobación</h2>

                <p><strong>Solicitante:</strong> {solicitante\_nombre}</p>

                <p><strong>Aprobado por supervisor:</strong> {supervisor\_nombre}</p>

                <p><strong>Motivo:</strong> {motivo}</p>

                <p><strong>Periodo solicitado:</strong> {fecha\_inicio} al {fecha\_fin}</p>

                <p><strong>Comentarios del supervisor:</strong> {comentarios or 'Ninguno'}</p>

                <hr>

                <p>Por favor revisa esta solicitud lo antes posible:</p>

                <a href="{enlace\_incidencia}" style="background-color: #4CAF50; color: white; padding: 10px 20px; text-align: center; text-decoration: none; display: inline-block; border-radius: 5px;">

                    Revisar Solicitud

                </a>

                <hr>

                <p><small>Este es un mensaje automático. Favor de no responder directamente.</small></p>

            </body>

            </html>

            """

            mensaje = **MIMEText**(cuerpo, 'html')

            mensaje['Subject'] = asunto

            mensaje['From'] = current\_app.config["MAIL\_USERNAME"]

            mensaje['To'] = correo\_gerente

*# Enviar el correo*

            with **smtplib**.**SMTP**(

                current\_app.config["MAIL\_SERVER"],

                current\_app.config["MAIL\_PORT"]

            ) as server:

                server.**starttls**()

                server.**login**(

                    current\_app.config["MAIL\_USERNAME"],

                    current\_app.config["MAIL\_PASSWORD"]

                )

                server.**send\_message**(mensaje)

            return True

        except **Exception** as e:

**print**(f"Error al enviar notificación al gerente: {**str**(e)}")

            return False

    def **obtener\_nombre\_usuario**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT nombre, apellidoPaterno

                FROM usuarios

                WHERE numNomina = ?

            """, (numNomina,))

            usuario = cursor.**fetchone**()

            return f"{usuario[0]} {usuario[1]}" if usuario else f"Nómina {numNomina}"

        except **Exception** as e:

**print**(f"Error al obtener nombre de usuario: {e}")

            return f"Nómina {numNomina}"

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_nombre\_usuario**(self, numNomina):

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT nombre + ' ' + ISNULL(apellidoPaterno, '')

                FROM usuarios

                WHERE numNomina = ?

            """, (numNomina,))

            nombre = cursor.**fetchone**()

            return nombre[0] if nombre else f"Usuario {numNomina}"

        except **Exception** as e:

**print**(f"Error al obtener nombre de usuario: {e}")

            return f"Usuario {numNomina}"

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_correo\_usuario**(self, numNomina):

        """Obtiene el correo electrónico de un usuario por su número de nómina"""

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("""

                SELECT correo\_electronico

                FROM usuarios

                WHERE numNomina = ?

            """, (numNomina,))

            resultado = cursor.**fetchone**()

            return resultado[0] if resultado else None

        except **Exception** as e:

**print**(f"Error al obtener correo del usuario: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

    def **get\_estatus\_incidencia**(self, idIncidencia):

        """Obtiene el estado actual de una incidencia"""

        conn = self.**get\_connection**()

        cursor = conn.**cursor**()

        try:

            cursor.**execute**("SELECT estatus FROM incidencias WHERE idIncidencia = ?", (idIncidencia,))

            resultado = cursor.**fetchone**()

            return resultado[0] if resultado else None

        except **Exception** as e:

**print**(f"Error al obtener estado de incidencia: {e}")

            return None

        finally:

            cursor.**close**()

            conn.**close**()

bienvenida.css

.pricing-card {

    border: 1px solid #ddd;

    border-radius: 10px;

    padding: 30px;

    text-align: center;

    transition: transform 0.3s ease;

    height: 250px; */\* Ajusta la altura según sea necesario \*/*

    display: flex;

    flex-direction: column;

    justify-content: center;

}

.pricing-card:hover {

    transform: **translateY**(-10px);

    background-color: #eeebeb;

    border: 2px solid black;

}

.pricing-card-header {

    font-size: 1.5rem;

    font-weight: bold;

}

.pricing-card-footer {

    margin-top: auto; */\* Empuja el contenido hacia abajo \*/*

}

.pricing-card {

    border-color: black;

}

.btn-custom {

    font-weight: bold;

    width: 100%; */\* Asegura que el botón ocupe todo el ancho \*/*

    margin-top: 10px; */\* Espacio entre la imagen y el botón \*/*

}

*/\* Animación de desvanecimiento \*/*

.fade-out {

    opacity: 0 **!important**;

    transition: opacity 0.5s ease-out **!important**;

}

.btn-close {

    margin-left: auto; */\* Empuja el botón a la derecha \*/*

}

.badge-nomina {

    font-size: 1.2rem;       */\* Tamaño de letra más grande \*/*

    padding: 0.5rem 1.2rem;  */\* Espaciado interno \*/*

    box-shadow: 0 2px 5px **rgba**(0,0,0,0.1); */\* Sombra suave \*/*

    transition: all 0.3s ease; */\* Efecto al pasar el mouse \*/*

}

.badge-nomina:hover {

    transform: **scale**(1.05);  */\* Efecto de escala al hover \*/*

}

*/\* Estilos para estados \*/*

.estado-aprobado {

    background-color: #28a745;

    color: white;

    font-weight: bold;

}

.estado-rechazado {

    background-color: #dc3545;

    color: white;

    font-weight: bold;

}

.estado-pendiente-supervisor {

    background-color: #ffc107;

    color: #212529;

    font-weight: bold;

}

.estado-pendiente-gerente {

    background-color: #17a2b8;

    color: white;

    font-weight: bold;

}

Login.css

*/\* CODIGO PARA login.html \*/*

\* {

  margin: 0;

  padding: 0;

  box-sizing: border-box;

}

body {

  font-family: Arial, sans-serif;

  background-color: #f3f4f6;

  display: flex;

  justify-content: center;

  align-items: center;

  min-height: 100vh; */\* Asegura el centrado vertical completo \*/*

  margin: 0;

}

.login-container {

  background-color: #ffffff;

  padding: 20px 30px;

  border-radius: 10px;

  box-shadow: 0 4px 6px **rgba**(0, 0, 0, 0.1);

  width: 100%;

  max-width: 400px;

}

.login-container h2 {

  text-align: center;

  margin-bottom: 20px;

  color: #333;

}

.form-group {

  margin-bottom: 15px;

}

.form-group label {

  display: block;

  font-weight: bold;

  margin-bottom: 5px;

}

.form-group input {

  width: 100%;

  padding: 10px;

  border: 1px solid #ccc;

  border-radius: 5px;

  font-size: 16px;

}

.form-group input:focus {

  border-color: #007bff;

  outline: none;

}

.btn {

  background-color: #007bff;

  color: #fff;

  padding: 10px 15px;

  border: none;

  border-radius: 5px;

  width: 100%;

  font-size: 16px;

  cursor: pointer;

}

.btn:hover {

  background-color: #0056b3;

}

.error-message {

  margin-bottom: 1rem;

  color: red;

  font-size: 0.9rem;

  margin-top: 10px;

  text-align: center;

}

*/\* Alertas personalizadas \*/*

    .alert-custom {

        border: none;

        border-radius: 10px;

        padding: 1rem 1.5rem;

        box-shadow: 0 2px 8px **rgba**(0,0,0,0.1);

        margin-bottom: 10px;

    }

    .alert-user\_not\_found {

        background: #fff3cd;

        color: #856404;

        border-left: 4px solid #ffc107;

    }

    .alert-wrong\_password {

        background: #f8d7da;

        color: #721c24;

        border-left: 4px solid #dc3545;

    }

*/\* Íconos \*/*

    .bi-person-x { color: #ffc107; }

    .bi-shield-lock { color: #dc3545; }

    .alert-custom {

      animation: slideIn 0.3s ease-out;

  }

  @keyframes slideIn {

      from { transform: **translateY**(-20px); opacity: 0; }

      to { transform: **translateY**(0); opacity: 1; }

  }

  .alert-custom {

    border: none;

    border-radius: 10px;

    padding: 1rem 1.5rem;

    box-shadow: 0 2px 8px **rgba**(0,0,0,0.1);

    transition: all 0.5s ease-out;

    opacity: 1;

    margin: 10px 0;

}

Agregar\_usuario.html

<!DOCTYPE *html*>

<html *lang*="es">

<head>

    <meta *charset*="UTF-8" />

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0" />

    <title>Agregar Usuario</title>

    <link *href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" *rel*="stylesheet" />

    <link *rel*="stylesheet" *href*="{{ url\_for('static', filename='css/bienvenida.css') }}" />

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div

*class*="container d-flex justify-content-between align-items-center gap-3"

        >

          <div *class*="d-flex align-items-center gap-3">

            <span *class*="badge bg-primary badge-nomina rounded-pill">

              <i *class*="bi bi-person-badge me-2"></i>

              {{ session['numNomina'] }}

            </span>

            <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

          </div>

          <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger"

            >Cerrar Sesión</a

          >

        </div>

      </header>

    <div *class*="container py-5">

        <h2>Agregar Usuario</h2>

*<!-- Mensajes de retroalimentación -->*

        {% if mensaje %}

        <div *class*="alert alert-info mt-3" *role*="alert">{{ mensaje }}</div>

        {% endif %}

*<!-- Formulario para agregar usuario -->*

        <form *method*="POST" *class*="row g-3">

            <div *class*="col-md-6">

                <div *class*="mb-3">

                    <label *for*="numNomina" *class*="form-label">Número de Nómina</label>

                    <input *type*="number" *class*="form-control" *id*="numNomina" *name*="numNomina" *required* />

                </div>

                <div *class*="mb-3">

                    <label *for*="nombre" *class*="form-label">Nombre</label>

                    <input *type*="text" *class*="form-control" *id*="nombre" *name*="nombre" *required* />

                </div>

                <div *class*="mb-3">

                    <label *for*="apellidoPaterno" *class*="form-label">Apellido Paterno</label>

                    <input *type*="text" *class*="form-control" *id*="apellidoPaterno" *name*="apellidoPaterno" />

                </div>

                <div *class*="mb-3">

                    <label *for*="apellidoMaterno" *class*="form-label">Apellido Materno</label>

                    <input *type*="text" *class*="form-control" *id*="apellidoMaterno" *name*="apellidoMaterno" />

                </div>

                <div *class*="mb-3">

                    <label *for*="username" *class*="form-label">Usuario</label>

                    <input *type*="text" *class*="form-control" *id*="username" *name*="username" *required* />

                </div>

                <div *class*="mb-3">

                    <label *for*="password" *class*="form-label">Contraseña</label>

                    <input *type*="password" *class*="form-control" *id*="password" *name*="password" *required* />

                </div>

            </div>

            <div *class*="col-md-6">

*<!-- Campo de Departamento -->*

                <div *class*="mb-3">

                    <label *for*="idDepartamento" *class*="form-label">Departamento</label>

                    <select *class*="form-select" *id*="idDepartamento" *name*="idDepartamento" *required*>

                        {% for departamento in departamentos %}

                            <option *value*="{{ departamento.idDepartamento }}">

                                {{ departamento.nombreDepartamento }}

                            </option>

                        {% endfor %}

                    </select>

                </div>

*<!-- Campo de Puesto -->*

                <div *class*="mb-3">

                    <label *for*="idPuesto" *class*="form-label">Puesto</label>

                    <select *class*="form-select" *id*="idPuesto" *name*="idPuesto" *required*>

*<!-- Los puestos se cargarán dinámicamente aquí -->*

                    </select>

                </div>

                <div *class*="mb-3">

                    <label *for*="idRol" *class*="form-label">Rol</label>

                    <select *class*="form-select" *id*="idRol" *name*="idRol" *required*>

                        {% for rol in roles %}

                            <option *value*="{{ rol.idRol }}">{{ rol.nombreRol }}</option>

                        {% endfor %}

                    </select>

                </div>

*<!-- Campo de Días de Vacaciones -->*

                <div *class*="mb-3">

                    <label *for*="diasVacaciones" *class*="form-label">Días de Vacaciones</label>

                    <input *type*="number" *class*="form-control" *id*="diasVacaciones" *name*="diasVacaciones" *required* />

                </div>

*<!-- Dentro del formulario de agregar\_usuario.html -->*

                <div *class*="mb-3">

                    <label *for*="correo\_electronico" *class*="form-label">Correo Electrónico</label>

                    <input *type*="email" *class*="form-control" *id*="correo\_electronico" *name*="correo\_electronico" *required*>

                </div>

                <div *class*="mb-3">

                    <label *for*="jefe\_directo" *class*="form-label">Jefe Directo</label>

                    <select *class*="form-select" *id*="jefe\_directo" *name*="jefe\_directo">

                        <option *value*="">Selecciona un jefe directo</option>

                        {% for usuario in usuarios %}

                            <option *value*="{{ usuario.numNomina }}">{{ usuario.nombre }} {{ usuario.apellidoPaterno }}</option>

                        {% endfor %}

                    </select>

                </div>

            </div>

        </form>

        <button *type*="submit" *class*="btn btn-success">Agregar Usuario</button>

        <a *href*="{{ url\_for('auth.bienvenida') }}" *class*="btn btn-secondary">Cancelar</a>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

*<!-- Al final del body -->*

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

    <script>

        document.**addEventListener**('DOMContentLoaded', function () {

            const selectDepartamento = document.**getElementById**('idDepartamento');

            const selectPuesto = document.**getElementById**('idPuesto');

*// Función para cargar los puestos según el departamento seleccionado*

            function **cargarPuestos**(idDepartamento) {

**fetch**(`/obtener\_puestos\_por\_departamento/${idDepartamento}`)

                    .**then**(response => response.**json**())

                    .**then**(data => {

*// Limpiar el select de puestos*

                        selectPuesto.innerHTML = '';

*// Agregar las opciones de puestos*

                        data.**forEach**(puesto => {

                            const option = document.**createElement**('option');

                            option.value = puesto.idPuesto;

                            option.textContent = puesto.nombrePuesto;

                            selectPuesto.**appendChild**(option);

                        });

                    })

                    .**catch**(error => console.**error**('Error al cargar puestos:', error));

            }

*// Cargar puestos cuando se seleccione un departamento*

            selectDepartamento.**addEventListener**('change', function () {

                const idDepartamento = this.value;

**cargarPuestos**(idDepartamento);

            });

*// Cargar puestos al cargar la página (según el primer departamento)*

            const idDepartamentoInicial = selectDepartamento.value;

            if (idDepartamentoInicial) {

**cargarPuestos**(idDepartamentoInicial);

            }

        });

    </script>

</body>

</html>

Bienvenida.html

<!DOCTYPE *html*>

<html *lang*="en">

  <head>

    <meta *charset*="UTF-8" />

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0" />

    <title>Bienvenida</title>

    <link

*href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"

*rel*="stylesheet"

    />

    <link

*rel*="stylesheet"

*href*="{{ url\_for('static', filename='css/bienvenida.css') }}"

    />

  </head>

  <body>

    <header *class*="bg-light py-3 mb-2">

      <div

*class*="container d-flex justify-content-between align-items-center gap-3"

      >

        <div *class*="d-flex align-items-center gap-3">

          <span *class*="badge bg-primary badge-nomina rounded-pill">

            <i *class*="bi bi-person-badge me-2"></i>

            {{ session['numNomina'] }}

          </span>

          <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

        </div>

        <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger"

          >Cerrar Sesión</a

        >

      </div>

    </header>

    <h1 *class*="text-center mt-4">MENU PRINCIPAL</h1>

    <div *class*="container py-0">

      <div *class*="row justify-content-center">

*<!-- Tarjeta de Agregar Usuario (solo para Admin, Gerente y Supervisor) -->*

          {% if session['rol'] == 'Admin' %}

          <div *class*="col-md-4 my-4">

              <div *class*="pricing-card agregar">

                  <div *class*="pricing-card-footer">

                      <img *src*="./static/img/Agregar Usuario.png" *alt*="agregar\_usuario" *style*="width: 100px; margin-bottom: 20px" /><br />

                      <a *href*="{{ url\_for('user.agregar\_usuario') }}" *class*="btn btn-success btn-custom">Agregar Usuario</a>

                  </div>

              </div>

          </div>

          {% endif %}

*<!-- Tarjeta de Ver Usuarios (solo para Admin) -->*

          {% if session['rol'] == 'Admin' %}

          <div *class*="col-md-4 my-4">

              <div *class*="pricing-card ver">

                  <div *class*="pricing-card-footer">

                      <img *src*="./static/img/Ver Usuario.png" *alt*="ver\_usuario" *style*="width: 100px; margin-bottom: 20px" /><br />

                      <a *href*="{{ url\_for('user.ver\_registros') }}" *class*="btn btn-primary btn-custom">Ver Usuarios</a>

                  </div>

              </div>

          </div>

          {% endif %}

*<!-- Tarjeta de Crear Incidencia Por Usuario (solo para Admin, Gerente y Supervisor) -->*

          {% if session['rol'] in ['Admin', 'Gerente', 'Supervisor'] %}

          <div *class*="col-md-4 my-4">

              <div *class*="pricing-card incidencias">

                  <div *class*="pricing-card-footer">

                      <img *src*="./static/img/incidencia\_usuario.png" *alt*="incidencias" *style*="width: 100px; margin-bottom: 20px" /><br />

                      <a *href*="{{ url\_for('user.incidencias\_usuario') }}" *class*="btn btn-warning btn-custom">Crear Incidencia Por Usuario</a>

                  </div>

              </div>

          </div>

          {% endif %}

*<!-- Tarjeta de Crear Incidencia (para todos los roles) -->*

          <div *class*="col-md-4 my-4">

              <div *class*="pricing-card incidencias">

                  <div *class*="pricing-card-footer">

                      <img *src*="./static/img/Incidencia.png" *alt*="incidencias" *style*="width: 100px; margin-bottom: 20px" /><br />

                      <a *href*="{{ url\_for('incidencias.crear\_incidencia\_usuario', numNomina=session['numNomina'], origen='bienvenida') }}" *class*="btn btn-warning btn-custom">Crear Incidencia</a>

                  </div>

              </div>

          </div>

*<!-- Tarjeta de Ver Mis Solicitudes (visible para todos los roles) -->*

          <div *class*="col-md-4 my-4">

            <div *class*="pricing-card ver-solicitudes">

                <div *class*="pricing-card-footer">

                    <img *src*="/static/img/mis\_incidencias.png" *alt*="ver\_solicitudes" *style*="width: 100px; margin-bottom: 20px" /><br />

                    <a *href*="{{ url\_for('incidencias.mis\_solicitudes') }}" *class*="btn btn-info btn-custom">Ver Mis Solicitudes</a>

                </div>

            </div>

          </div>

*<!-- Tarjeta de Incidencias Recibidas (solo para Usuario, Supervisor y Gerente) -->*

          {% if session['rol'] in ['Admin', 'Supervisor', 'Gerente'] %}

          <div *class*="col-md-4 my-4">

            <div *class*="pricing-card incidencias-recibidas">

                <div *class*="pricing-card-footer">

                    <img *src*="/static/img/incidencias\_recibidas.png" *alt*="incidencias\_recibidas" *style*="width: 100px; margin-bottom: 20px" /><br />

                    <a *href*="{{ url\_for('incidencias.solicitudes\_recibidas') }}" *class*="btn btn-warning btn-custom">Incidencias Recibidas</a>

                </div>

            </div>

          </div>

          {% endif %}

      </div>

      </div>

    </div>

*<!-- Mensajes Flash -->*

    <div *id*="flash-container">

      {% with messages = get\_flashed\_messages(with\_categories=true) %} {% if

      messages %} {% for category, message in messages %}

      <div

*class*="alert alert-{{ 'danger' if category == 'error' else 'success' }} alert-autoclose d-flex justify-content-between align-items-center mt-3"

*role*="alert"

      >

        <span>{{ message }}</span>

*<!-- Mensaje a la izquierda -->*

        <button

*type*="button"

*class*="btn-close"

*data-bs-dismiss*="alert"

*aria-label*="Close"

        ></button>

      </div>

      {% endfor %} {% endif %} {% endwith %}

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

*<!-- Al final del body -->*

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

    <script>

      document.**addEventListener**(

        "DOMContentLoaded",

        function () {

          console.**log**("DOM cargado"); *// Depuración 1*

          const alerts = document.**querySelectorAll**(".alert-autoclose");

          console.**log**("Alertas encontradas:", alerts.length); *// Depuración 2*

          alerts.**forEach**((alert, index) => {

            console.**log**(`Iniciando temporizador para alerta ${index}`); *// Depuración 3*

**setTimeout**(() => {

              alert.classList.**add**("fade-out");

**setTimeout**(() => {

                console.**log**(`Alerta ${index} eliminada`); *// Depuración 4*

                alert.**remove**();

              }, 500);

            }, 2000);

          });

        },

        { once: true }

      ); *// Evita múltiples ejecuciones*

    </script>

  </body>

</html>

Editar\_usuario.html

<!DOCTYPE *html*>

<html *lang*="es">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <title>Editar Usuario</title>

    <link *href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" *rel*="stylesheet">

    <link *rel*="stylesheet" *href*="{{ url\_for('static', filename='css/bienvenida.css') }}">

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div *class*="container d-flex justify-content-between align-items-center gap-3">

            <div *class*="d-flex align-items-center gap-3">

                <span *class*="badge bg-primary badge-nomina rounded-pill">

                    <i *class*="bi bi-person-badge me-2"></i>

                    {{ session['numNomina'] }}

                </span>

                <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

            </div>

            <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger">Cerrar Sesión</a>

        </div>

    </header>

    <div *class*="container py-5">

        <h1 *class*="h3 mb-3">Editar Usuario</h1>

*<!-- Mensajes de error/éxito -->*

        {% with messages = get\_flashed\_messages(with\_categories=true) %}

            {% if messages %}

                {% for category, message in messages %}

                    <div *class*="alert alert-{{ 'danger' if category == 'error' else 'success' }} alert-dismissible fade show" *role*="alert">

                        {{ message }}

                        <button *type*="button" *class*="btn-close" *data-bs-dismiss*="alert" *aria-label*="Close"></button>

                    </div>

                {% endfor %}

            {% endif %}

        {% endwith %}

*<!-- Formulario de edición -->*

        <form *method*="POST" *class*="row g-3">

            <div *class*="col-md-6">

                <div *class*="mb-3">

                    <label *for*="numNomina" *class*="form-label">Número de Nómina</label>

                    <input *type*="number" *class*="form-control" *id*="numNomina" *name*="numNomina" *value*="{{ usuario.numNomina }}" *readonly* />

                </div>

                <div *class*="mb-3">

                    <label *for*="nombre" *class*="form-label">Nombre</label>

                    <input *type*="text" *class*="form-control" *id*="nombre" *name*="nombre" *value*="{{ usuario.nombre }}" *required* />

                </div>

                <div *class*="mb-3">

                    <label *for*="apellidoPaterno" *class*="form-label">Apellido Paterno</label>

                    <input *type*="text" *class*="form-control" *id*="apellidoPaterno" *name*="apellidoPaterno" *value*="{{ usuario.apellidoPaterno }}" />

                </div>

                <div *class*="mb-3">

                    <label *for*="apellidoMaterno" *class*="form-label">Apellido Materno</label>

                    <input *type*="text" *class*="form-control" *id*="apellidoMaterno" *name*="apellidoMaterno" *value*="{{ usuario.apellidoMaterno }}" />

                </div>

                <div *class*="mb-3">

                    <label *for*="username" *class*="form-label">Usuario</label>

                    <input *type*="text" *class*="form-control" *id*="username" *name*="username" *value*="{{ usuario.username }}" *required* />

                </div>

                <div *class*="mb-3">

                    <label *for*="password" *class*="form-label">Contraseña</label>

                    <input *type*="text" *class*="form-control" *id*="password" *name*="password" *value*="{{ usuario.password }}" *required* />

                </div>

            </div>

            <div *class*="col-md-6">

*<!-- Campo de Departamento -->*

            <div *class*="mb-3">

                <label *for*="idDepartamento" *class*="form-label">Departamento</label>

                <select *class*="form-select" *id*="idDepartamento" *name*="idDepartamento" *required*>

                    {% for departamento in departamentos %}

                        <option *value*="{{ departamento.idDepartamento }}" *{%* *if* *departamento.idDepartamento* == usuario.idDepartamento *%}selected{%* *endif* *%}*>

                            {{ departamento.nombreDepartamento }}

                        </option>

                    {% endfor %}

                </select>

            </div>

*<!-- Campo de Puesto -->*

            <div *class*="mb-3">

                <label *for*="idPuesto" *class*="form-label">Puesto</label>

                <select *class*="form-select" *id*="idPuesto" *name*="idPuesto" *required*>

                    {% for puesto in puestos %}

                        <option *value*="{{ puesto.idPuesto }}" *{%* *if* *puesto.idPuesto* == usuario.idPuesto *%}selected{%* *endif* *%}*>

                            {{ puesto.nombrePuesto }}

                        </option>

                    {% endfor %}

                </select>

            </div>

*<!-- Campo de Días de Vacaciones -->*

            <div *class*="mb-3">

                <label *for*="diasVacaciones" *class*="form-label">Días de Vacaciones</label>

                <input *type*="number" *class*="form-control" *id*="diasVacaciones" *name*="diasVacaciones" *value*="{{ usuario.diasVacaciones }}" *readonly* />

            </div>

*<!-- Campo de Rol -->*

            <div *class*="mb-3">

                <label *for*="idRol" *class*="form-label">Rol</label>

                <select *class*="form-select" *id*="idRol" *name*="idRol" *{%* *if* *session[*'rol'] *!*= 'Admin' *%}disabled{%* *endif* *%}* *required*>

                    {% for rol in roles %}

                        <option *value*="{{ rol.idRol }}" *{%* *if* *rol.idRol* == usuario.idRol *%}selected{%* *endif* *%}*>

                            {{ rol.nombreRol }}

                        </option>

                    {% endfor %}

                </select>

            </div>

*<!-- Dentro del formulario de editar\_usuario.html -->*

            <div *class*="mb-3">

                <label *for*="correo\_electronico" *class*="form-label">Correo Electrónico</label>

                <input *type*="email" *class*="form-control" *id*="correo\_electronico" *name*="correo\_electronico" *value*="{{ usuario.correo\_electronico }}" *required*>

            </div>

            <div *class*="mb-3">

                <label *for*="jefe\_directo" *class*="form-label">Jefe Directo</label>

                <select *class*="form-select" *id*="jefe\_directo" *name*="jefe\_directo">

                    <option *value*="">Selecciona un jefe directo</option>

                    {% for usuario\_jefe in usuarios %}

                        <option *value*="{{ usuario\_jefe.numNomina }}" *{%* *if* *usuario\_jefe.numNomina* == usuario.jefe\_directo *%}selected{%* *endif* *%}*>

                            {{ usuario\_jefe.nombre }} {{ usuario\_jefe.apellidoPaterno }}

                        </option>

                    {% endfor %}

                </select>

            </div>

            </div>

            <div *class*="mt-4">

                <button *type*="submit" *class*="btn btn-success">Guardar Cambios</button>

                <a *href*="{{ url\_for('user.ver\_registros') }}" *class*="btn btn-secondary">Cancelar</a>

            </div>

        </form>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

*<!-- Al final del body -->*

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

    <script>

        document.**addEventListener**('DOMContentLoaded', function () {

            const selectDepartamento = document.**getElementById**('idDepartamento');

            const selectPuesto = document.**getElementById**('idPuesto');

*// Función para cargar los puestos según el departamento seleccionado*

            function **cargarPuestos**(idDepartamento) {

**fetch**(`/obtener\_puestos\_por\_departamento/${idDepartamento}`)

                    .**then**(response => response.**json**())

                    .**then**(data => {

*// Limpiar el select de puestos*

                        selectPuesto.innerHTML = '';

*// Agregar las opciones de puestos*

                        data.**forEach**(puesto => {

                            const option = document.**createElement**('option');

                            option.value = puesto.idPuesto;

                            option.textContent = puesto.nombrePuesto;

                            selectPuesto.**appendChild**(option);

                        });

*// Seleccionar el puesto actual del usuario*

                        const puestoActual = "{{ usuario.idPuesto }}";  *// Asegúrate de que esto esté correctamente pasado desde el controlador*

                        if (puestoActual) {

                            selectPuesto.value = puestoActual;

                            console.**log**(`Puesto actual seleccionado: ${puestoActual}`);  *// Depuración*

                        }

                    })

                    .**catch**(error => console.**error**('Error al cargar puestos:', error));

            }

*// Cargar puestos cuando se seleccione un departamento*

            selectDepartamento.**addEventListener**('change', function () {

                const idDepartamento = this.value;

**cargarPuestos**(idDepartamento);

            });

*// Cargar puestos al cargar la página (según el primer departamento)*

            const idDepartamentoInicial = selectDepartamento.value;

            if (idDepartamentoInicial) {

**cargarPuestos**(idDepartamentoInicial);

            }

        });

    </script>

*<!-- Script para ocultar mensajes automáticamente -->*

     <script >

        document.**addEventListener**('DOMContentLoaded', function() {

    const alerts = document.**querySelectorAll**('.alert');

    alerts.**forEach**(alert => {

**setTimeout**(() => {

            alert.style.opacity = '0';

**setTimeout**(() => alert.**remove**(), 500);

        }, 2000);

    });

});

     </script>

</body>

</html>

Incidencias.html

<!DOCTYPE *html*>

<html *lang*="en">

  <head>

    <meta *charset*="UTF-8" />

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0" />

    <title>Control de Ausencia y Salidas de Planta</title>

    <link

*href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"

*rel*="stylesheet"

    />

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

    <link

*rel*="stylesheet"

*href*="{{ url\_for('static', filename='css/bienvenida.css') }}"

    />

  </head>

  <body>

    <header *class*="bg-light py-3 mb-2">

      <div

*class*="container d-flex justify-content-between align-items-center gap-3"

      >

        <div *class*="d-flex align-items-center gap-3">

          <span *class*="badge bg-primary badge-nomina rounded-pill">

            <i *class*="bi bi-person-badge me-2"></i>

            {{ session['numNomina'] }}

          </span>

          <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

        </div>

        <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger"

          >Cerrar Sesión</a

        >

      </div>

    </header>

    <div *class*="container mt-5">

*<!-- Encabezado de la página -->*

      <div *class*="card mb-2 shadow">

        <div *class*="card-header bg-primary">

          <h5 *class*="mb-0 text-white">Solicitud de Incidencia</h5>

        </div>

        <div *class*="card-body">

          <div *class*="row">

*<!-- Logo a la izquierda -->*

            <div *class*="col-md-4 d-flex align-items-center">

              <img

*src*="{{ url\_for('static', filename='img/Logo-transparente.ico') }}"

*alt*="logo\_tohken"

*class*="img-fluid"

*style*="width: 75px"

              />

            </div>

*<!-- TIO-702.01.03/B en el centro -->*

            <div

*class*="col-md-4 d-flex align-items-center justify-content-center text-center"

            >

              <p *class*="mb-0" *style*="font-weight: bold; font-size: 16px">

                TIO-702.01.03/B <br />

                Control de Ausencia y Salidas de Planta

              </p>

            </div>

*<!-- Control de Ausencia y Salidas de Planta / Edición a la derecha -->*

            <div

*class*="col-md-4 d-flex align-items-center justify-content-end text-center"

            >

              <div *class*="text-end">

                <p

*class*="text-muted mb-0"

*style*="font-weight: bold; font-size: 16px"

                >

                  Edición 1ra. <br />

                  Fecha: 10-Ene-20

                </p>

              </div>

            </div>

          </div>

        </div>

      </div>

*<!-- Formulario de incidencia -->*

      <div *class*="card shadow">

        <div *class*="card-body">

          <form

*method*="POST"

*action*="{{ url\_for('incidencias.crear\_incidencia') }}"

          >

*<!-- Información del solicitante -->*

              <div *class*="row mb-3" *style*="font-size: 14px">

                <div *class*="col-md-3">

                  <label *for*="nombre" *class*="form-label">Nombre(s) del Solicitante</label>

                  <input *type*="text" *class*="form-control" *id*="nombre" *name*="nombre" *value*="{{ nombre }}" *readonly* />

              </div>

              <div *class*="col-md-3">

                  <label *for*="apellido\_paterno" *class*="form-label">Apellido Paterno</label>

                  <input *type*="text" *class*="form-control" *id*="apellido\_paterno" *name*="apellido\_paterno" *value*="{{ apellido\_paterno }}" *readonly* />

              </div>

              <div *class*="col-md-3">

                  <label *for*="apellido\_materno" *class*="form-label">Apellido Materno</label>

                  <input *type*="text" *class*="form-control" *id*="apellido\_materno" *name*="apellido\_materno" *value*="{{ apellido\_materno }}" *readonly* />

              </div>

              <div *class*="col-md-3">

                  <label *for*="fecha\_solicitud" *class*="form-label">Fecha de Solicitud</label>

                  <input *type*="date" *class*="form-control" *id*="fecha\_solicitud" *name*="fecha\_solicitud" *value*="{{ fecha\_solicitud }}" *readonly* />

              </div>

*<!-- Campo de nombre del puesto -->*

              <div *class*="col-md-3" *style*="margin-top: 15px;">

                <label *for*="puesto" *class*="form-label">Puesto</label>

                <input *type*="text" *class*="form-control" *id*="puesto" *name*="puesto" *value*="{{ nombre\_puesto }}" *readonly* />

              </div>

*<!-- Campo de Número de Nómina -->*

              <div *class*="col-md-3" *style*="margin-top: 15px;">

                  <label *for*="no\_nomina" *class*="form-label">No. Nómina</label>

                  <input *type*="number" *class*="form-control" *id*="no\_nomina" *name*="no\_nomina" *value*="{{ no\_nomina }}" *readonly* />

              </div>

*<!-- Campo de nombre del departamento -->*

            <div *class*="col-md-3" *style*="margin-top: 15px;">

              <label *for*="departamento" *class*="form-label">Departamento</label>

              <input *type*="text" *class*="form-control" *id*="departamento" *name*="departamento" *value*="{{ nombre\_departamento }}" *readonly* />

            </div>

*<!-- Campo de días de vacaciones restantes -->*

            <div *class*="col-md-3" *style*="margin-top: 15px;">

              <label *for*="dias\_vacaciones" *class*="form-label">Días de Vacaciones Restantes</label>

              <input *type*="number" *class*="form-control" *id*="dias\_vacaciones" *name*="dias\_vacaciones" *value*="{{ dias\_vacaciones }}" *readonly* />

            </div>

          </div>

*<!-- Vacaciones restantes, Fecha de inicio y Fecha de fin -->*

          <div *class*="row mb-2" *style*="font-size: 14px;">

            <div *class*="col-md-3">

              <label *for*="motivo\_ausencia" *class*="form-label">Motivo</label>

              <select

*class*="form-select"

*id*="motivo\_ausencia"

*name*="motivo"

*style*="font-size: 14px"

*required*

              >

                <option *value*="goce\_sueldo">

                  Permiso con goce de sueldo

                </option>

                <option *value*="sin\_goce\_sueldo">

                  Permiso sin goce de sueldo

                </option>

                <option *value*="vacaciones">Vacaciones</option>

                <option *value*="olvido\_gafete">Olvido de gafete</option>

                <option *value*="tiempo\_x\_tiempo">Tiempo x Tiempo</option>

              </select>

            </div>

*<!-- Campo de fecha de inicio -->*

            <div *class*="col-md-2">

              <label *for*="fecha\_inicio" *class*="form-label">Fecha de Inicio</label>

              <input *type*="date" *class*="form-control" *id*="fecha\_inicio" *name*="fecha\_inicio" *required* />

            </div>

*<!-- Campo de fecha de fin -->*

            <div *class*="col-md-2">

              <label *for*="fecha\_fin" *class*="form-label">Fecha de Fin</label>

              <input *type*="date" *class*="form-control" *id*="fecha\_fin" *name*="fecha\_fin" *required* />

            </div>

*<!-- Campo de No. de Días (no editable) -->*

            <div *class*="col-md-1">

              <label *for*="num\_dias" *class*="form-label">No. de Días</label>

              <input *type*="number" *class*="form-control" *id*="num\_dias" *name*="num\_dias" *readonly* />

            </div>

*<!-- Observaciones -->*

          <div *class*="mb-4" *style*="font-size: 14px; margin-top: 15px;">

            <label *for*="comentarios" *class*="form-label">Observaciones</label>

            <textarea *class*="form-control" *id*="comentarios" *name*="comentarios" *rows*="4" *required*></textarea>

            <small *class*="form-text text-muted">Máximo 250 caracteres.</small>

          </div>

*<!-- Botones de acción -->*

          <div *class*="d-flex justify-content-between">

            <form *method*="POST" *action*="{{ url\_for('incidencias.crear\_incidencia') }}">

*<!-- Campos del formulario -->*

              <button *type*="submit" *class*="btn btn-success">Enviar Solicitud</button>

          </form>

            <a *href*="{{

                url\_for('auth.bienvenida') if origen == 'bienvenida'

                else url\_for('user.incidencias\_usuario') if origen == 'incidencias\_usuario'

                else url\_for('user.ver\_registros')

            }}" *class*="btn btn-danger">Cancelar</a>

          </div>

          </div>

          </form>

        </div>

      </div>

    </div>

*<!-- Scripts de Bootstrap -->*

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.min.js"></script>

    <script>

      document.**addEventListener**('DOMContentLoaded', function () {

*// Obtener los valores del departamento y puesto del usuario*

          const nombreDepartamento = document.**getElementById**('nombre\_departamento').value;

          const nombrePuesto = document.**getElementById**('nombre\_puesto').value;

*// Seleccionar el departamento y puesto en los selectores*

          const selectDepartamento = document.**getElementById**('departamento');

          const selectPuesto = document.**getElementById**('puesto');

          for (let option of selectDepartamento.options) {

              if (option.text === nombreDepartamento) {

                  option.selected = true;

                  break;

              }

          }

          for (let option of selectPuesto.options) {

              if (option.text === nombrePuesto) {

                  option.selected = true;

                  break;

              }

          }

      });

      </script>

      <script>

        document.**addEventListener**('DOMContentLoaded', function () {

            const fechaInicio = document.**getElementById**('fecha\_inicio');

            const fechaFin = document.**getElementById**('fecha\_fin');

            const numDias = document.**getElementById**('num\_dias');

*// Función para calcular el número de días*

            function **calcularDias**() {

                if (fechaInicio.value && fechaFin.value) {

                    const inicio = new **Date**(fechaInicio.value);

                    const fin = new **Date**(fechaFin.value);

                    const diferencia = fin - inicio;

                    const dias = Math.**floor**(diferencia / (1000 \* 60 \* 60 \* 24)) + 1; *// +1 para incluir el día de inicio*

                    numDias.value = dias;

                } else {

                    numDias.value = '';

                }

            }

*// Escuchar cambios en las fechas*

            fechaInicio.**addEventListener**('change', **calcularDias**);

            fechaFin.**addEventListener**('change', **calcularDias**);

        });

        </script>

  </body>

</html>

Incidencias\_usuario.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8" />

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0" />

    <title>Incidencias de Usuario</title>

    <link *href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" *rel*="stylesheet" />

    <link *rel*="stylesheet" *href*="{{ url\_for('static', filename='css/bienvenida.css') }}" />

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div *class*="container d-flex justify-content-between align-items-center gap-3">

            <div *class*="d-flex align-items-center gap-3">

                <span *class*="badge bg-primary badge-nomina rounded-pill">

                    <i *class*="bi bi-person-badge me-2"></i>

                    {{ session['numNomina'] }}

                </span>

                <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

            </div>

            <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger">Cerrar Sesión</a>

        </div>

    </header>

    <div *class*="container py-5">

        <h1 *class*="h3 mb-3">Incidencias de Usuario</h1>

*<!-- Formulario de búsqueda -->*

        <form *method*="POST" *class*="mb-4">

            <div *class*="input-group">

                <input *type*="text" *name*="numNomina" *class*="form-control" *placeholder*="Buscar por Número de Nómina" *value*="{{ busqueda if busqueda else '' }}"/>

                <button *class*="btn btn-success" *type*="submit">Buscar</button>

            </div>

        </form>

*<!-- Tabla de registros -->*

        <div *class*="table-responsive">

            <table *class*="table table-striped table-bordered">

                <thead>

                    <tr>

                        <th>Número de Nómina</th>

                        <th>Nombre</th>

                        <th>Apellido Paterno</th>

                        <th>Apellido Materno</th>

                        <th>Departamento</th>

                        <th>Puesto</th>

                        <th>Días de Vacaciones</th>

                        <th *class*="text-center">Acciones</th>

                    </tr>

                </thead>

                <tbody>

                    {% if registros %}

                        {% for registro in registros %}

                            <tr>

                                <td *class*="align-middle border border-1">{{ registro.numNomina }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombre }}</td>

                                <td *class*="align-middle border border-1">{{ registro.apellidoPaterno }}</td>

                                <td *class*="align-middle border border-1">{{ registro.apellidoMaterno }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombreDepartamento }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombrePuesto }}</td>

                                <td *class*="align-middle border border-1">{{ registro.diasVacaciones }}</td>

                                <td *class*="align-middle border border-1">

                                    <div *class*="d-flex justify-content-center">

*<!-- Botón de crear incidencia -->*

                                        <a *href*="{{ url\_for('incidencias.crear\_incidencia\_usuario', numNomina=registro.numNomina, origen='incidencias\_usuario') }}" *class*="btn btn-primary">

                                            Crear Incidencia

                                        </a>

                                    </div>

                                </td>

                            </tr>

                        {% endfor %}

                    {% else %}

                        <tr>

                            <td *colspan*="8" *class*="text-center">No se encontraron registros</td>

                        </tr>

                    {% endif %}

                </tbody>

            </table>

        </div>

*<!-- Botón de regresar -->*

        <div *class*="container d-flex justify-content-between align-items-center ps-0">

            <a *href*="{{ url\_for('auth.bienvenida') }}" *class*="btn btn-secondary mt-3">Regresar</a>

        </div>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link *rel*="stylesheet" *href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css" />

</body>

</html>

Login.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <title>Login Form</title>

    <link *rel*="stylesheet" *href*="{{ url\_for('static', filename='css/login.css') }}">

</head>

<body>

    <div *class*="login-container">

        <h2>Login</h2>

        <form *action*="{{ url\_for('auth.do\_login') }}" *method*="POST">

*<!-- Campo de entrada -->*

            <div *class*="form-group">

                <label *for*="login\_input">Usuario o Número de Nómina</label>

                <input

*type*="text"

*class*="form-control"

*id*="login\_input"

*name*="login\_input"

*placeholder*="Ej: nombre.apellido o 12345"

*required*

                >

            </div>

*<!-- Campo de contraseña -->*

            <div *class*="form-group">

                <label *for*="password">Contraseña</label>

                <input

*type*="password"

*class*="form-control"

*id*="password"

*name*="password"

*placeholder*="Ingresa tu contraseña"

*required*

                >

            </div>

*<!-- Dentro del formulario -->*

            {% with messages = get\_flashed\_messages(with\_categories=true) %}

                {% if messages %}

                    {% for category, message in messages %}

                        <div *class*="alert alert-custom alert-{{ category }} d-flex align-items-center gap-3">

                            {% if category == "user\_not\_found" %}

                                <i *class*="bi bi-person-x fs-4"></i>

                            {% elif category == "wrong\_password" %}

                                <i *class*="bi bi-shield-lock fs-4"></i>

                            {% endif %}

                            <span *class*="flex-grow-1">{{ message }}</span>

                            <button *type*="button" *class*="btn-close" *data-bs-dismiss*="alert"></button>

                        </div>

                    {% endfor %}

                {% endif %}

            {% endwith %}

            <button *type*="submit" *class*="btn btn-primary mt-3 w-100">Iniciar Sesión</button>

        </form>

    </div>

</body>

</html>

Mis\_solicitudes.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8"/>

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0"/>

    <title>Mis Solicitudes</title>

    <link

*href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"

*rel*="stylesheet"

    />

    <link

*rel*="stylesheet"

*href*="{{ url\_for('static', filename='css/bienvenida.css') }}"

    />

  </head>

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div *class*="container d-flex justify-content-between align-items-center gap-3">

            <div *class*="d-flex align-items-center gap-3">

                <span *class*="badge bg-primary badge-nomina rounded-pill">

                    <i *class*="bi bi-person-badge me-2"></i>

                    {{ session['numNomina'] }}

                </span>

                <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

            </div>

            <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger">Cerrar Sesión</a>

        </div>

    </header>

    <div *class*="container py-5">

        <h1 *class*="h3">Mis Solicitudes</h1>

        <div *class*="mb-3">

            <a *href*="{{ url\_for('incidencias.mis\_solicitudes', orden='asc') }}" *class*="btn btn-outline-primary btn-sm">Ascendente</a>

            <a *href*="{{ url\_for('incidencias.mis\_solicitudes', orden='desc') }}" *class*="btn btn-outline-primary btn-sm">Descendente</a>

        </div>

        <div *class*="table-responsive">

            <table *class*="table table-striped table-bordered">

                <thead>

                    <tr>

                        <th *class*="text-center">ID</th>

                        <th *class*="text-center">Solicitante</th>

                        <th *class*="text-center">Fecha</th>

                        <th *class*="text-center">Motivo</th>

                        <th *class*="text-center">Estado</th>

                        <th *class*="text-center">Acciones</th>

                    </tr>

                </thead>

                <tbody>

                    {% for solicitud in solicitudes %}

                    <tr>

                        <td *class*="text-center">{{ solicitud.idIncidencia }}</td>

                        <td *class*="text-center">{{ solicitud.numNomina\_solicitante }}</td>

                        <td *class*="text-center">{{ solicitud.fecha\_solicitud }}</td>

                        <td *class*="text-center">{{ solicitud.motivo }}</td>

                        <td *class*="text-center

                            {{ 'bg-success text-white' if solicitud.estatus == 'Aprobada' else

                            'bg-danger text-white' if solicitud.estatus == 'Rechazada' else

                            'bg-warning text-dark' if solicitud.estatus == 'Pendiente Supervisor' else

                            'bg-info text-white' if solicitud.estatus == 'Pendiente Gerente' else

                            'bg-secondary text-white' }}">

                            {{ solicitud.estatus }}

                        </td>

                        <td *class*="text-center">

                            <a *href*="{{ url\_for('incidencias.ver\_incidencia',

                                idIncidencia=solicitud.idIncidencia,

                                origen='incidencias\_recibidas') }}"

*class*="btn btn-primary">

                                Ver Solicitud

                            </a>

                        </td>

                    </tr>

                    {% endfor %}

                </tbody>

            </table>

        </div>

*<!-- Botón de regresar -->*

        <div *class*="mt-4">

            <a *href*="{{ url\_for('auth.bienvenida') }}" *class*="btn btn-secondary">Regresar</a>

        </div>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

</body>

</html>

Solicitudes\_recibidas.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8"/>

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0"/>

    <title>Incidencias Recibidas</title>

    <link

*href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"

*rel*="stylesheet"

    />

    <link

*rel*="stylesheet"

*href*="{{ url\_for('static', filename='css/bienvenida.css') }}"

    />

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div *class*="container d-flex justify-content-between align-items-center gap-3">

            <div *class*="d-flex align-items-center gap-3">

                <span *class*="badge bg-primary badge-nomina rounded-pill">

                    <i *class*="bi bi-person-badge me-2"></i>

                    {{ session['numNomina'] }}

                </span>

                <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

            </div>

            <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger">Cerrar Sesión</a>

        </div>

    </header>

    <div *class*="container py-5">

        <h1 *class*="h3 mb-3">Incidencias Recibidas</h1>

        <div *class*="d-flex justify-content-between align-items-center mb-3">

            <div *class*="mb-3">

                <a *href*="{{ url\_for('incidencias.solicitudes\_recibidas', orden='asc') }}" *class*="btn btn-outline-primary btn-sm">Ascendente</a>

                <a *href*="{{ url\_for('incidencias.solicitudes\_recibidas', orden='desc') }}" *class*="btn btn-outline-primary btn-sm">Descendente</a>

            </div>

        </div>

        <div *class*="table-responsive">

            <table *class*="table table-striped table-bordered">

                <thead>

                    <tr>

                        <th *class*="text-center">ID</th>

                        <th *class*="text-center">Solicitante</th>

                        <th *class*="text-center">Fecha</th>

                        <th *class*="text-center">Motivo</th>

                        <th *class*="text-center">Estado</th>

                        <th *class*="text-center">Acciones</th>

                    </tr>

                </thead>

                <tbody>

                    {% for solicitud in solicitudes %}

                    <tr>

                        <td *class*="text-center">{{ solicitud.idIncidencia }}</td>

                        <td *class*="text-center">{{ solicitud.numNomina\_solicitante }}</td>

                        <td *class*="text-center">{{ solicitud.fecha\_solicitud }}</td>

                        <td *class*="text-center">{{ solicitud.motivo }}</td>

                        <td *class*="text-center

                            {{ 'bg-success text-white' if solicitud.estatus == 'Aprobada' else

                            'bg-danger text-white' if solicitud.estatus == 'Rechazada' else

                            'bg-warning text-dark' if solicitud.estatus == 'Pendiente Supervisor' else

                            'bg-info text-white' if solicitud.estatus == 'Pendiente Gerente' else

                            'bg-secondary text-white' }}">

                            {{ solicitud.estatus }}

                        </td>

                        <td *class*="text-center">

                            <a *href*="{{ url\_for('incidencias.ver\_incidencia',

                                idIncidencia=solicitud.idIncidencia,

                                origen='incidencias\_recibidas') }}"

*class*="btn btn-primary">

                                Ver Incidencia

                            </a>

                            {% if solicitud.estatus in ['Pendiente Supervisor', 'Pendiente Gerente'] %}

                                {% set puede\_aprobar = (session['rol'] == 'Supervisor' and solicitud.estatus == 'Pendiente Supervisor') or

                                                      (session['rol'] == 'Gerente' and solicitud.estatus == 'Pendiente Gerente') %}

                            {% endif %}

                        </td>

                    </tr>

                    {% endfor %}

                </tbody>

            </table>

        </div>

*<!-- Botón de regresar -->*

        <div *class*="mt-4">

            <a *href*="{{ url\_for('auth.bienvenida') }}" *class*="btn btn-secondary">Regresar</a>

        </div>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

</body>

</html>

Ver\_incidencia.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8"/>

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0"/>

    <title>Ver Incidencia</title>

    <link

*href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"

*rel*="stylesheet"

    />

    <link

*rel*="stylesheet"

*href*="{{ url\_for('static', filename='css/bienvenida.css') }}"

    />

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div *class*="container d-flex justify-content-between align-items-center gap-3">

            <div *class*="d-flex align-items-center gap-3">

                <span *class*="badge bg-primary badge-nomina rounded-pill">

                    <i *class*="bi bi-person-badge me-2"></i>

                    {{ session['numNomina'] }}

                </span>

                <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

            </div>

            <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger">Cerrar Sesión</a>

        </div>

    </header>

*<!-- Mensaje para incidencias ya procesadas -->*

            {% if incidencia.estatus in ['Aprobada', 'Rechazada'] %}

              <div *class*="alert alert-info text-center mb-0">

                  Esta incidencia ya ha sido procesada y no puede ser modificada.

                  {% if incidencia.aprobado\_por\_supervisor %}

                      <strong>Supervisor:</strong> {{ obtener\_nombre\_usuario(incidencia.aprobado\_por\_supervisor) }}

                  {% endif %}

                  {% if incidencia.aprobado\_por\_gerente %}

                      <strong>Gerente:</strong> {{ obtener\_nombre\_usuario(incidencia.aprobado\_por\_gerente) }}

                  {% endif %}

              </div>

          {% endif %}

    <div *class*="container py-2">

        <div *class*="container mt-0">

*<!-- Encabezado de la página -->*

            <div *class*="card mb-2 shadow">

              <div *class*="card-header bg-primary">

                <h5 *class*="mb-0 text-white">Solicitud de Incidencia</h5>

              </div>

              <div *class*="card-body">

                <div *class*="row">

*<!-- Logo a la izquierda -->*

                  <div *class*="col-md-4 d-flex align-items-center">

                    <img

*src*="{{ url\_for('static', filename='img/Logo-transparente.ico') }}"

*alt*="logo\_tohken"

*class*="img-fluid"

*style*="width: 75px"

                    />

                  </div>

*<!-- TIO-702.01.03/B en el centro -->*

                  <div

*class*="col-md-4 d-flex align-items-center justify-content-center text-center"

                  >

                    <p *class*="mb-0" *style*="font-weight: bold; font-size: 16px">

                      TIO-702.01.03/B <br />

                      Control de Ausencia y Salidas de Planta

                    </p>

                  </div>

*<!-- Control de Ausencia y Salidas de Planta / Edición a la derecha -->*

                  <div

*class*="col-md-4 d-flex align-items-center justify-content-end text-center"

                  >

                    <div *class*="text-end">

                      <p

*class*="text-muted mb-0"

*style*="font-weight: bold; font-size: 16px"

                      >

                        Edición 1ra. <br />

                        Fecha: 10-Ene-20

                      </p>

                    </div>

                  </div>

                </div>

              </div>

            </div>

        <div *class*="card shadow">

            <div *class*="card-body">

                <div *class*="row mb-3" *style*="font-size: 14px">

                    <div *class*="col-md-3">

                        <label *for*="nombre" *class*="form-label">Nombre(s) del Solicitante</label>

                        <input *type*="text" *class*="form-control" *id*="nombre" *name*="nombre" *value*="{{incidencia.nombre\_solicitante}}" *readonly* />

                    </div>

                    <div *class*="col-md-3">

                        <label *for*="apellido\_paterno" *class*="form-label">Apellido Paterno</label>

                        <input *type*="text" *class*="form-control" *id*="apellido\_paterno" *name*="apellido\_paterno" *value*="{{ incidencia.apellido\_paterno }}" *readonly* />

                    </div>

                    <div *class*="col-md-3">

                        <label *for*="apellido\_materno" *class*="form-label">Apellido Materno</label>

                        <input *type*="text" *class*="form-control" *id*="apellido\_materno" *name*="apellido\_materno" *value*="{{ incidencia.apellido\_materno }}" *readonly* />

                    </div>

                    <div *class*="col-md-3">

                        <label *for*="fecha\_solicitud" *class*="form-label">Fecha de Solicitud</label>

                        <input *type*="date" *class*="form-control" *id*="fecha\_solicitud" *name*="fecha\_solicitud" *value*="{{ incidencia.fecha\_solicitud }}" *readonly* />

                    </div>

*<!-- Campo de nombre del puesto -->*

                    <div *class*="col-md-3" *style*="margin-top: 15px;">

                        <label *for*="puesto" *class*="form-label">Puesto</label>

                        <input *type*="text" *class*="form-control" *id*="puesto" *name*="puesto" *value*="{{ incidencia.puesto }}" *readonly* />

                    </div>

*<!-- Campo de Número de Nómina -->*

                    <div *class*="col-md-3" *style*="margin-top: 15px;">

                        <label *for*="no\_nomina" *class*="form-label">No. Nómina</label>

                        <input *type*="number" *class*="form-control" *id*="no\_nomina" *name*="no\_nomina" *value*="{{ incidencia.numNomina\_solicitante }}" *readonly* />

                    </div>

*<!-- Campo de nombre del departamento -->*

                    <div *class*="col-md-3" *style*="margin-top: 15px;">

                        <label *for*="departamento" *class*="form-label">Departamento</label>

                        <input *type*="text" *class*="form-control" *id*="departamento" *name*="departamento" *value*="{{ incidencia.departamento }}" *readonly* />

                    </div>

                    <div *class*="col-md-3" *style*="margin-top: 15px;">

                        <label *for*="dias\_vacaciones" *class*="form-label">Días de Vacaciones Restantes</label>

                        <input *type*="number" *class*="form-control" *id*="dias\_vacaciones" *name*="dias\_vacaciones" *value*="{{ incidencia.dias\_vacaciones }}" *readonly* />

                    </div>

                </div>

                <div *class*="row mb-2" *style*="font-size: 14px;">

                    <div *class*="col-md-3">

                        <label *for*="motivo\_ausencia" *class*="form-label">Motivo</label>

                        <input *type*="text" *class*="form-control" *id*="motivo" *name*="motivo" *value*="{{ incidencia.motivo }}" *readonly* />

                    </div>

                    <div *class*="col-md-2">

                        <label *for*="fecha\_inicio" *class*="form-label">Fecha de Inicio</label>

                        <input *type*="text" *class*="form-control" *id*="fecha\_inicio" *name*="fecha\_inicio" *value*="{{ incidencia.fecha\_inicio }}" *readonly* />

                    </div>

                    <div *class*="col-md-2">

                        <label *for*="fecha\_inicio" *class*="form-label">Fecha de Fin</label>

                        <input *type*="text" *class*="form-control" *id*="fecha\_fin" *name*="fecha\_fin" *value*="{{ incidencia.fecha\_fin }}" *readonly* />

                    </div>

                    <div *class*="col-md-1">

                        <label *for*="num\_dias" *class*="form-label">No. de Días</label>

                        <input *type*="number" *class*="form-control" *id*="num\_dias" *name*="num\_dias" *value*="{{ incidencia.num\_dias }}" *readonly* />

                    </div>

                </div>

*<!-- Observaciones -->*

                <div *class*="mb-4" *style*="font-size: 14px; margin-top: 15px;">

                    <label *for*="comentarios" *class*="form-label">Observaciones</label>

                    <textarea *class*="form-control" *id*="comentarios" *name*="comentarios" *placeholder*="{{ incidencia.observaciones }}" *rows*="4" *readonly*></textarea>

                </div>

                <p *class*="card-text"><strong>Estatus:</strong> {{ incidencia.estatus }}</p>

            </div>

        </div>

*<!-- Tarjetas de Aprobación en una fila -->*

        <div *class*="row mt-3 mb-0">

*<!-- Tarjeta de Supervisor -->*

          <div *class*="col-md-6">

              <div *class*="card h-100">

                  <div *class*="card-header info bg-secondary text-white">

                      {% if incidencia.aprobado\_por\_supervisor %}

                          {% if incidencia.estatus == 'Rechazada' and incidencia.aprobado\_por\_supervisor %}

                              Rechazo del Supervisor

                          {% else %}

                              Aprobación del Supervisor

                          {% endif %}

                      {% else %}

                          Aprobación del Supervisor

                      {% endif %}

                  </div>

                  <div *class*="card-body">

                      <p><strong>

                          {% if incidencia.aprobado\_por\_supervisor %}

                              {% if incidencia.estatus == 'Rechazada' and incidencia.aprobado\_por\_supervisor %}

                                  Rechazado por:

                              {% else %}

                                  Aprobado por:

                              {% endif %}

                          {% else %}

                              Estado:

                          {% endif %}

                      </strong>

                      {% if incidencia.aprobado\_por\_supervisor %}

                          {{ obtener\_nombre\_usuario(incidencia.aprobado\_por\_supervisor) }}

                      {% else %}

                          Pendiente

                      {% endif %}</p>

                      <p><strong>Fecha:</strong> {{ incidencia.fecha\_aprobacion\_supervisor or 'Pendiente' }}</p>

                      <p><strong>Comentarios:</strong> {{ incidencia.comentarios\_supervisor or 'Sin comentarios' }}</p>

                  </div>

              </div>

          </div>

*<!-- Tarjeta de Gerente -->*

          <div *class*="col-md-6">

              <div *class*="card h-100">

                  <div *class*="card-header info bg-secondary text-white">

                      {% if incidencia.aprobado\_por\_gerente %}

                          {% if incidencia.estatus == 'Rechazada' and incidencia.aprobado\_por\_gerente %}

                              Rechazo del Gerente

                          {% else %}

                              Aprobación del Gerente

                          {% endif %}

                      {% else %}

                          Aprobación del Gerente

                      {% endif %}

                  </div>

                  <div *class*="card-body">

                      <p><strong>

                          {% if incidencia.aprobado\_por\_gerente %}

                              {% if incidencia.estatus == 'Rechazada' and incidencia.aprobado\_por\_gerente %}

                                  Rechazado por:

                              {% else %}

                                  Aprobado por:

                              {% endif %}

                          {% else %}

                              Estado:

                          {% endif %}

                      </strong>

                      {% if incidencia.aprobado\_por\_gerente %}

                          {{ obtener\_nombre\_usuario(incidencia.aprobado\_por\_gerente) }}

                      {% else %}

                          Pendiente

                      {% endif %}</p>

                      <p><strong>Fecha:</strong> {{ incidencia.fecha\_aprobacion\_gerente or 'Pendiente' }}</p>

                      <p><strong>Comentarios:</strong> {{ incidencia.comentarios\_gerente or 'Sin comentarios' }}</p>

                  </div>

              </div>

          </div>

        </div>

*<!-- Botones de acción -->*

        <div *class*="mt-2 d-flex gap-2">

          {% if incidencia.estatus == 'Pendiente Supervisor' and session['numNomina'] == incidencia.jefe\_directo %}

            <form *method*="POST" *action*="{{ url\_for('incidencias.procesar\_incidencia', idIncidencia=incidencia.idIncidencia) }}" *class*="d-flex gap-2">

                <input *type*="hidden" *name*="comentarios" *id*="comentariosRechazo">

                <button *type*="submit" *name*="accion" *value*="aprobar" *class*="btn btn-success"

*onclick*="document.**getElementById**('comentariosRechazo').value = **prompt**('Comentarios (opcional):') || ''">

                    Aprobar (enviar a gerente)

                </button>

                <button *type*="submit" *name*="accion" *value*="rechazar" *class*="btn btn-danger"

*onclick*="document.**getElementById**('comentariosRechazo').value = **prompt**('Motivo del rechazo (requerido):') || ''; return !!document.**getElementById**('comentariosRechazo').value;">

                    Rechazar Definitivamente

                </button>

            </form>

          {% elif incidencia.estatus == 'Pendiente Gerente' and session['numNomina'] == incidencia.gerente\_responsable %}

*<!-- Botones para gerente -->*

              <form *method*="POST" *action*="{{ url\_for('incidencias.procesar\_incidencia', idIncidencia=incidencia.idIncidencia) }}" *class*="d-flex gap-2">

                  <input *type*="hidden" *name*="comentarios" *id*="comentariosGerente">

                  <button *type*="submit" *name*="accion" *value*="aprobar" *class*="btn btn-success"

*onclick*="document.**getElementById**('comentariosGerente').value = **prompt**('Comentarios (opcional):') || ''">

                      Aprobar como Gerente

                  </button>

                  <button *type*="submit" *name*="accion" *value*="rechazar" *class*="btn btn-danger"

*onclick*="document.**getElementById**('comentariosGerente').value = **prompt**('Razón del rechazo:') || ''">

                      Rechazar

                  </button>

              </form>

          {% endif %}

*<!-- Botón de regresar siempre visible -->*

            <a *href*="javascript:history.back()"

*class*="btn btn-secondary">

            Regresar

 </a>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link

*rel*="stylesheet"

*href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"

    />

</body>

</html>

Ver\_registros.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8" />

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0" />

    <title>Ver Registros</title>

    <link *href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" *rel*="stylesheet" />

    <link *rel*="stylesheet" *href*="{{ url\_for('static', filename='css/bienvenida.css') }}" />

</head>

<body>

    <header *class*="bg-light py-3 mb-2">

        <div *class*="container d-flex justify-content-between align-items-center gap-3">

            <div *class*="d-flex align-items-center gap-3">

                <span *class*="badge bg-primary badge-nomina rounded-pill">

                    <i *class*="bi bi-person-badge me-2"></i>

                    {{ session['numNomina'] }}

                </span>

                <h1 *class*="h3 mb-0 text-nowrap">{{ session['user'] }}</h1>

            </div>

            <a *href*="{{ url\_for('auth.logout') }}" *class*="btn btn-danger">Cerrar Sesión</a>

        </div>

    </header>

    <div *class*="container py-5">

        <h1 *class*="h3 mb-3">Registros</h1>

        <form *method*="POST" *class*="mb-4">

            <div *class*="input-group">

                <input *type*="text" *name*="numNomina" *class*="form-control" *placeholder*="Buscar por Número de Nómina" *value*="{{ busqueda if busqueda else '' }}"/>

                <button *class*="btn btn-success" *type*="submit">Buscar</button>

            </div>

        </form>

        <div *class*="table-responsive">

            <table *class*="table table-striped table-bordered">

                <thead>

                    <tr>

                        <th>Número de Nómina</th>

                        <th>Nombre</th>

                        <th>Apellido Paterno</th>

                        <th>Apellido Materno</th>

                        <th>Usuario</th>

                        <th>Departamento</th>

                        <th>Puesto</th>

                        <th>Días de Vacaciones</th>

                        <th>Rol</th>

                        <th>Correo Electrónico</th>

                        <th>Jefe Directo</th>

                        <th *class*="text-center">Acciones</th>

                    </tr>

                </thead>

                <tbody>

                    {% if registros %}

                        {% for registro in registros %}

                            <tr>

                                <td *class*="align-middle border border-1">{{ registro.numNomina }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombre }}</td>

                                <td *class*="align-middle border border-1">{{ registro.apellidoPaterno }}</td>

                                <td *class*="align-middle border border-1">{{ registro.apellidoMaterno }}</td>

                                <td *class*="align-middle border border-1">{{ registro.username }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombreDepartamento }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombrePuesto }}</td>

                                <td *class*="align-middle border border-1">{{ registro.diasVacaciones }}</td>

                                <td *class*="align-middle border border-1">{{ registro.nombreRol }}</td>

                                <td *class*="align-middle border border-1">{{ registro.correo\_electronico }}</td>

                                <td *class*="align-middle border border-1">

                                    {% if registro.nombre\_jefe %}

                                        {{ registro.nombre\_jefe }} {{ registro.apellido\_jefe }}

                                    {% else %}

                                        Sin jefe asignado

                                    {% endif %}

                                </td>

                                <td *class*="align-middle border border-1">

                                    <div *class*="d-flex justify-content-center">

*<!-- Botón de Editar -->*

                                        <a *href*="{{ url\_for('user.editar\_usuario', numNomina=registro.numNomina) }}" *class*="btn btn-warning me-2">

                                            Editar Usuario

                                        </a>

*<!-- Botón de Borrar (solo visible para Admin) -->*

                                        {% if session['rol'] == 'Admin' %}

                                            <form *method*="POST" *action*="{{ url\_for('user.borrar\_usuario', numNomina=registro.numNomina) }}">

                                                <button *type*="submit" *class*="btn btn-danger">Borrar Usuario</button>

                                            </form>

                                        {% endif %}

                                    </div>

                                </td>

                            </tr>

                        {% endfor %}

                    {% else %}

                        <tr>

                            <td *colspan*="9" *class*="text-center">No se encontraron registros</td>

                        </tr>

                    {% endif %}

                </tbody>

            </table>

        </div>

        <div *class*="container d-flex justify-content-between align-items-center ps-0">

            <a *href*="{{ url\_for('auth.bienvenida') }}" *class*="btn btn-secondary mt-3">Regresar</a>

        </div>

    </div>

    <script *src*="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>

    <script *src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js"></script>

    <link *rel*="stylesheet" *href*="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css" />

</body>

</html>

\_\_init\_\_.py

from **flask** import **Flask**

from **app**.**controllers**.**auth\_controller** import auth\_bp

from **app**.**controllers**.**user\_controller** import user\_bp

from **app**.**controllers**.**incidencias\_controller** import incidencias\_bp

from **app**.**controllers**.**user\_controller** import user\_bp

def **create\_app**():

    app = **Flask**(

        \_\_name\_\_,

        template\_folder="views/templates",  *# ¡Ruta personalizada!*

        static\_folder="views/static"        *# Ruta para archivos estáticos*

    )

    app.secret\_key = "tu\_clave\_secreta"

    app.config["MAIL\_SERVER"] = "smtp.gmail.com"  *# Ejemplo para Gmail*

    app.config["MAIL\_PORT"] = 587

    app.config["MAIL\_USE\_TLS"] = True

    app.config["MAIL\_USERNAME"] = "escaner.ttm@tohken.mx"

    app.config["MAIL\_PASSWORD"] = "flst rwdn vgam ouxh"  *# No la expongas públicamente*

    app.**register\_blueprint**(auth\_bp)

    app.**register\_blueprint**(user\_bp)

    app.**register\_blueprint**(incidencias\_bp)

    return app

run.py

from **flask** import session

from **app** import **create\_app**

from **datetime** import **datetime**

app = **create\_app**()

from **app**.**models**.**user\_model** import **UserModel**

user\_model = **UserModel**()

*# Registrar la función en Jinja2*

**@app.context\_processor**

def **utility\_processor**():

    return {

        'obtener\_nombre\_usuario': user\_model.**obtener\_nombre\_usuario**,

        'current\_user\_rol': lambda: session.**get**('rol', None)  *# Función adicional útil*

    }

**@app.template\_filter**('format\_datetime')

def **format\_datetime\_filter**(value, format="%d/%m/%Y %H:%M"):

    if value is None:

        return ""

    if **isinstance**(value, **str**):

        value = **datetime**.**strptime**(value, '%Y-%m-%d %H:%M:%S')

    return value.**strftime**(**format**)

if \_\_name\_\_ == '\_\_main\_\_':

    app.**run**(debug=True)

esta es la estructura de mi base de datos

CREATE TABLE [dbo].[departamento\_puesto](

[idDepartamento] [int] NOT NULL,

[idPuesto] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[idDepartamento] ASC,

[idPuesto] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[departamentos] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[departamentos](

[idDepartamento] [int] IDENTITY(1,1) NOT NULL,

[nombreDepartamento] [nvarchar](100) NOT NULL,

PRIMARY KEY CLUSTERED

(

[idDepartamento] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],

UNIQUE NONCLUSTERED

(

[nombreDepartamento] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[incidencias] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[incidencias](

[idIncidencia] [int] IDENTITY(1,1) NOT NULL,

[numNomina\_solicitante] [int] NOT NULL,

[nombre\_solicitante] [nvarchar](100) NOT NULL,

[apellido\_paterno] [nvarchar](100) NOT NULL,

[apellido\_materno] [nvarchar](100) NOT NULL,

[fecha\_solicitud] [date] NOT NULL,

[puesto] [nvarchar](100) NOT NULL,

[departamento] [nvarchar](100) NOT NULL,

[dias\_vacaciones] [int] NOT NULL,

[motivo] [nvarchar](100) NOT NULL,

[fecha\_inicio] [date] NOT NULL,

[fecha\_fin] [date] NOT NULL,

[num\_dias] [int] NOT NULL,

[observaciones] [nvarchar](500) NULL,

[estatus] [nvarchar](50) NOT NULL,

[aprobado\_por\_supervisor] [int] NULL,

[fecha\_aprobacion\_supervisor] [datetime] NULL,

[aprobado\_por\_gerente] [int] NULL,

[fecha\_aprobacion\_gerente] [datetime] NULL,

[comentarios\_supervisor] [nvarchar](500) NULL,

[comentarios\_gerente] [nvarchar](500) NULL,

[jefe\_directo] [int] NOT NULL,

[gerente\_responsable] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[idIncidencia] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[jefes] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[jefes](

[idJefe] [int] NOT NULL,

[numNomina] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[idJefe] ASC,

[numNomina] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[motivos] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[motivos](

[idMotivo] [int] IDENTITY(1,1) NOT NULL,

[nombreMotivo] [nvarchar](100) NOT NULL,

PRIMARY KEY CLUSTERED

(

[idMotivo] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[puestos] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[puestos](

[idPuesto] [int] IDENTITY(1,1) NOT NULL,

[nombrePuesto] [nvarchar](100) NOT NULL,

PRIMARY KEY CLUSTERED

(

[idPuesto] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],

UNIQUE NONCLUSTERED

(

[nombrePuesto] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],

CONSTRAINT [uq\_nombrePuesto] UNIQUE NONCLUSTERED

(

[nombrePuesto] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[roles] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[roles](

[idRol] [int] IDENTITY(1,1) NOT NULL,

[nombreRol] [nvarchar](50) NOT NULL,

PRIMARY KEY CLUSTERED

(

[idRol] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],

UNIQUE NONCLUSTERED

(

[nombreRol] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[usuario\_rol] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[usuario\_rol](

[numNomina] [int] NOT NULL,

[idRol] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[numNomina] ASC,

[idRol] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[usuarios] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[usuarios](

[numNomina] [int] NOT NULL,

[nombre] [nvarchar](100) NOT NULL,

[apellidoPaterno] [nvarchar](100) NULL,

[apellidoMaterno] [nvarchar](100) NULL,

[idDepartamento] [int] NULL,

[idPuesto] [int] NULL,

[diasVacaciones] [int] NULL,

[correo\_electronico] [nvarchar](255) NULL,

[jefe\_directo] [int] NULL,

PRIMARY KEY CLUSTERED

(

[numNomina] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[vacaciones] Script Date: 02/04/2025 03:50:59 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[vacaciones](

[idVacaciones] [int] IDENTITY(1,1) NOT NULL,

[numNomina] [int] NOT NULL,

[diasVacaciones] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[idVacaciones] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[incidencias] ADD DEFAULT ('Pendiente Supervisor') FOR [estatus]

GO

ALTER TABLE [dbo].[usuarios] ADD DEFAULT ((0)) FOR [diasVacaciones]

GO

ALTER TABLE [dbo].[credenciales] WITH CHECK ADD FOREIGN KEY([numNomina])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[departamento\_puesto] WITH CHECK ADD FOREIGN KEY([idDepartamento])

REFERENCES [dbo].[departamentos] ([idDepartamento])

GO

ALTER TABLE [dbo].[departamento\_puesto] WITH CHECK ADD FOREIGN KEY([idPuesto])

REFERENCES [dbo].[puestos] ([idPuesto])

GO

ALTER TABLE [dbo].[departamento\_puesto] WITH CHECK ADD CONSTRAINT [fk\_departamento] FOREIGN KEY([idDepartamento])

REFERENCES [dbo].[departamentos] ([idDepartamento])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[departamento\_puesto] CHECK CONSTRAINT [fk\_departamento]

GO

ALTER TABLE [dbo].[departamento\_puesto] WITH CHECK ADD CONSTRAINT [fk\_puesto] FOREIGN KEY([idPuesto])

REFERENCES [dbo].[puestos] ([idPuesto])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[departamento\_puesto] CHECK CONSTRAINT [fk\_puesto]

GO

ALTER TABLE [dbo].[incidencias] WITH CHECK ADD CONSTRAINT [FK\_incidencias\_gerente] FOREIGN KEY([gerente\_responsable])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[incidencias] CHECK CONSTRAINT [FK\_incidencias\_gerente]

GO

ALTER TABLE [dbo].[incidencias] WITH CHECK ADD CONSTRAINT [FK\_incidencias\_supervisor] FOREIGN KEY([jefe\_directo])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[incidencias] CHECK CONSTRAINT [FK\_incidencias\_supervisor]

GO

ALTER TABLE [dbo].[jefes] WITH CHECK ADD FOREIGN KEY([idJefe])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[jefes] WITH CHECK ADD FOREIGN KEY([numNomina])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[usuario\_rol] WITH CHECK ADD FOREIGN KEY([idRol])

REFERENCES [dbo].[roles] ([idRol])

GO

ALTER TABLE [dbo].[usuario\_rol] WITH CHECK ADD FOREIGN KEY([numNomina])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[usuarios] WITH CHECK ADD CONSTRAINT [fk\_jefe\_directo] FOREIGN KEY([jefe\_directo])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

ALTER TABLE [dbo].[usuarios] CHECK CONSTRAINT [fk\_jefe\_directo]

GO

ALTER TABLE [dbo].[usuarios] WITH CHECK ADD CONSTRAINT [fk\_usuario\_departamento] FOREIGN KEY([idDepartamento])

REFERENCES [dbo].[departamentos] ([idDepartamento])

GO

ALTER TABLE [dbo].[usuarios] CHECK CONSTRAINT [fk\_usuario\_departamento]

GO

ALTER TABLE [dbo].[usuarios] WITH CHECK ADD CONSTRAINT [fk\_usuario\_puesto] FOREIGN KEY([idPuesto])

REFERENCES [dbo].[puestos] ([idPuesto])

GO

ALTER TABLE [dbo].[usuarios] CHECK CONSTRAINT [fk\_usuario\_puesto]

GO

ALTER TABLE [dbo].[vacaciones] WITH CHECK ADD FOREIGN KEY([numNomina])

REFERENCES [dbo].[usuarios] ([numNomina])

GO

USE [master]

GO

ALTER DATABASE [Prueba\_10] SET READ\_WRITE

GO