Módulo Documentos Internos

- Lista de categorias - documentos (1) - [getInternalDocuments]
- Guardar documento (1) - [saveInternalDocuments]
- Examen Medico (1) - [obtain]

Lista de categorias - documentos (1) - [getInternalDocuments]

? Descripción

Obtiene **todos los documentos internos del ERP**, los agrupa por categoría y devuelve cuáles corresponden a un puesto específico (si aplica).

Este servicio combina información procedente de:

- `tabDocumentos Internos`
- `tabDocumentos especificos` (relación hijo por puesto)

También determina dinámicamente qué archivo debe adjuntarse para la categoría **IPERC**, según el puesto del trabajador.

□□ Es un servicio de consulta sin paginación.
□□ No requiere parámetros en el body (solo el parámetro $puesto en la URL).

? Endpoint

GET /internal-documents/{puesto}

- `{puesto}` → Nombre del puesto (Designation).
- Si no se envía o es null, el servicio igualmente devolverá las categorías, pero sin personalizar la categoría IPERC.

? Seguridad

Requiere autenticación válida vía `dbErp` usando el token del ERP.
El servicio accede directamente a vistas y tablas internas del ERP.

? Flujo del Servicio

1. Consulta documentos internos

Ejecuta la sentencia SQL:

```
SELECT
    di.name,
    di.nombre_del_archivo,
    di.categoria,
    di.campo_para_adjuntar,
    de.puesto,
    de.adjuntar
FROM `tabDocumentos Internos` di
LEFT JOIN `tabDocumentos especificos` de
    ON di.name = de.parent
```

Si falla la consulta → devuelve error 500 del servicio.

---

2. Normaliza categorías

Si un documento no tiene categoría, se asigna por defecto: `categoria = "POLITICAS"`

---

3. Agrupa documentos por categoría

Genera una estructura como:

```
{
  "POLITICAS": [...],
  "IPERC": [...],
  "MOF": [...],
  ...
}
```

---

4. Caso especial: categoría IPERC

Para IPERC:

- Se toma el adjunto por defecto → `campo_para_adjuntar`
- Si el puesto recibido coincide con algún registro hijo (`tabDocumentos especificos`)
  → se usa el archivo personalizado (`adjuntar`)

Ejemplo lógica:

```
adjunto = campo_para_adjuntar_default

foreach item in detalles:
    if item.puesto == puesto AND item.adjuntar != "":
        adjunto = item.adjuntar
```

---

5. Construye la respuesta final

La respuesta incluye:

- Lista de categorías
- Lista agrupada con sus documentos y el archivo a adjuntar según la lógica

---

? Response 200 – Ejemplo Real


```
{
  "valor": true,
  "msn": "Lista de categorias generado correctamente",
  "data": {
    "categorias": [
      "POLITICAS",
      "IPERC",
      "MOF"
    ],
    "categorias_registro": {
      "POLITICAS": [
        {
          "name": "DOC-001",
          "nombre_del_archivo": "reglamento_interno.pdf",
          "campo_para_adjuntar": "/files/reglamento_interno.pdf"
        }
      ],
      "IPERC": [
        {
          "name": "IPERC-001",
          "nombre_del_archivo": "iperc_almacen.pdf",
          "campo_para_adjuntar": "/files/iperc_almacen.pdf"
        }
      ]
    }
  }
}
```

? Posibles errores

1. Error de servicio ERP

```
{
  "valor": false,
  "msn": "Error de servicio",
  "data": [...]
}
```

2. Error interno

```
{
  "valor": false,
  "msn": "Error de servidor",
  "data": "<mensaje>"
}
```

? Estructuras de datos utilizadas

Documentos Internos (`tabDocumentos Internos`)

Campos utilizados:

```
{
  "name": "string",
  "nombre_del_archivo": "string",
  "categoria": "string|null",
  "campo_para_adjuntar": "string"
}
```

Documentos Específicos (`tabDocumentos especificos`)

Campos utilizados:

```
{
  "puesto": "string|null",
  "adjuntar": "string|null"
}
```

? Pseudocódigo del servicio

```
sql = SELECT documentos internos + documentos especificos

docs = dbErp(sql)

foreach doc in docs:
    if categoria is null:
        categoria = "POLITICAS"

grouped = group docs by categoria

foreach categoria in grouped:
    group by name (padre)

    if categoria == 'IPERC':
        adjunto = campo_para_adjuntar_default
        if puesto matches child:
            adjunto = child.adjuntar
    else:
        adjunto = campo_para_adjuntar

return {
  categorias: keys(grouped),
  categorias_registro: grouped
}
```

Guardar documento (1) - [saveInternalDocuments]

? Descripción

Registra internamente un documento aceptado o gestionado por un usuario dentro de una terminal específica.

Este servicio guarda un log en la tabla **documentos_internos**, permitiendo rastrear:

- Qué usuario aceptó o gestionó un documento.
- Desde qué terminal lo realizó.
- Qué tipo de documento fue.
- En qué fecha y hora se registró.

Es utilizado para auditoría y trazabilidad dentro del sistema.

? Endpoint

POST `/save-internal-documents`

? Parámetros (Request Body)

```
{
  "usuario": "string",
  "terminal": "string",
  "tipo": "string"
}
```

Campos:

Campo	Tipo	Requerido	Descripción

usuario	string	✔ Sí	ID o username del usuario que realiza la acción
terminal	string	✔ Sí	Código de la terminal donde se ejecuta la acción
tipo	string	□ No	Tipo de documento o acción registrada

? Seguridad

No requiere token de ERP.
Acceso interno del backend usando conexión directa a la base de datos **mysql2**.

? Flujo del Servicio (resumen)

1. Valida que **usuario** y **terminal** existan en la solicitud.
2. Establece zona horaria **America/Lima**.
3. Inserta un registro en la tabla `documentos_internos` con:
   - usuario
   - terminal
   - tipo
   - fecha actual
4. Si el registro falla, retorna un error.
5. Si todo es exitoso, responde confirmación de inserción.

? Base de Datos Utilizada

Tabla: `documentos_internos` (BD: mysql2)

Campo	Tipo	Descripción
usuario	varchar	Usuario que registra
terminal	varchar	Terminal desde donde registra
tipo	varchar	Tipo de documento
fecha	datetime	Fecha y hora del registro

? Pseudocódigo

```
if usuario vacío → retornar error “te olvidaste el usuario”
if terminal vacío → retornar error "te olvidaste la terminal"

fecha = now()

insert into documentos_internos (usuario, terminal, tipo, fecha)

if insert falla → retornar "Ocurrió un error"
else → retornar "Insertado con éxito"
```

? Respuestas del Servicio

? **200 – Inserción exitosa**

```
{
  "valor": true,
  "msg": "Insertado con éxito"
}
```

? Error: Falta usuario

```
{
  "valor": false,
  "msg": "te olvidaste el usuario"
}
```

? Error: Falta terminal

```
{
  "valor": false,
  "msg": "te olvidaste la terminal"
}
```

? Error al insertar

```
{
  "valor": false,
  "msg": "Ocurrio un error"
}
```

? Ejemplo completo de Request

```
{
  "usuario": "user@example.com",
  "terminal": "236",
  "tipo": "descarga_politicas"
}
```

Examen Medico (1) - [obtain]

? Descripción

*Este servicio permite **consultar los resultados del examen EMO** (Evaluación Médica Ocupacional) de un empleado utilizando su **DNI (passport_number)**.*

El servicio valida que el DNI haya sido enviado, busca la información correspondiente mediante el método interno `getExamenEmo()` y retorna los resultados si existen.

Es un servicio de consulta rápida y directa.

? Endpoint

POST /obtain

? Request Body

```
{
  "passport_number": "12345678"
}
```

Campos:

Campo	Tipo	Obligatorio	Descripción
passport_number	string	✔	DNI del empleado para consultar su examen EMO

? Seguridad

Este servicio **no requiere autenticación externa**, pero su funcionamiento depende de un método interno:

- `getExamenEmo(passport_number)` → Consulta la información del examen EMO.

? Flujo del Servicio (Explicación)

1. **Validación inicial del DNI**
   - Si el parámetro `passport_number` no está presente, el servicio responde con error.
2. **Consulta del examen EMO**
   - Se llama al método interno `getExamenEmo($passport_number)`.
3. **Validación del resultado**
   - Si el método no retorna datos, se responde con un mensaje indicando que no se encontraron resultados.
4. **Respuesta exitosa**
   - Si sí existen resultados, se retorna el contenido del examen EMO.

? Response 200 – Ejemplos

?? Caso exitoso

```
{
  "valor": true,
  "msn": "Se encontraron resultados",
  "data": {
    "resultado": "Apto",
    "fecha": "2025-01-12",
    "observaciones": "Ninguna"
  }
}
```

? Error: DNI no enviado

```
{
  "valor": false,
  "msn": "El dni del empelado es obligatorio"
}
```

? Error: Sin resultados EMO

```
{
  "valor": false,
  "msn": "No se encontraron resultados"
}
```

? Posibles Errores

Código	Motivo	Descripción
400	DNI faltante	No se envió el `passport_number`
404	Sin resultados	El empleado no tiene examen EMO registrado
500	Error en método interno	Fallo en `getExamenEmo()`

? Schemas

Request Schema

```
{
  "passport_number": "string"
}
```

Response Schema (éxito)

```
{
  "valor": true,
  "msn": "Se encontraron resultados",
  "data": { ...examen_emo }
}
```

Response Schema (error)

```
{
  "valor": false,
  "msn": "Descripción del error"
}
```

? Lógica en pseudocódigo

```
if !passport_number:
    return error("El dni del empleado es obligatorio")

examen = getExamenEmo(passport_number)

if !examen:
    return error("No se encontraron resultados")

return success(examen)
```