

# Documento de Requisitos del Servicio de Procesamiento y Subida de Archivos

## 1. Descripción General

El servicio debe recibir datos de entrada, descargar un archivo remoto, optimizarlo según su tipo y subirlo al bucket de Supabase utilizando endpoints específicos. Al finalizar, debe devolver el enlace público del recurso cargado. El servicio deberá alojarse preferentemente en Firebase.

---

## 2. Entrada del Servicio

El servicio recibirá una solicitud con un body en formato JSON con la siguiente estructura:

```
{
  "linkArchivo": "string",
  "tipoDeArchivo": "string",
  "signedUrl": "string"
}
```

### 2.1. Parámetros

- **linkArchivo:** URL del archivo a descargar.
  - **tipoDeArchivo:** Identifica el tipo de documento procesado. Se usa para construir el `public_url` final.
  - **signedUrl:** URL firmada proporcionada por Supabase para subir el archivo optimizado.. Parámetros
  - **tipoDeArchivo:** String que indica el tipo de archivo a procesar. Define el endpoint para la subida. Valores válidos:
    - `CSF`
    - `ComprobanteDomiciliario`
    - `Ticket`
- 

## 3. Descarga y Optimización de Archivos

El servicio debe descargar el archivo contenido en `linkArchivo`.

### 3.1. Tipos soportados

- PDF
- JPG
- JPEG

### 3.2. Reglas de optimización

Técnica de optimización a libre elección. Buscar reducir lo máximo posible el tamaño sin comprometer altamente la legibilidad.

## 4. Subida de Archivos a Supabase

El servicio debe usar **exclusivamente** la signed URL recibida en el body para subir el archivo optimizado.

### 4.1. Mecanismo de subida

- El archivo optimizado será enviado mediante método PUT a la signed URL.
- No se debe generar endpoints internos según tipo de archivo.

### 4.2. Uso de tipoDeArchivo

El valor `tipoDeArchivo` no determina el destino de subida, pero sí se usa para construir o clasificar la `public_url` generada.

### 4.3. Inserción en el bucket

- La subida se realiza directamente usando la signed URL recibida.
- El nombre del archivo ya debe estar contenido en la signed URL.

## 5. Obtención de Enlace Público.

Obtención de Enlace Público Una vez subido el archivo:

- El servicio debe generar el enlace público según el valor en `tipoDeArchivo`.
- Debe regresar una respuesta JSON con dicho enlace.

### 5.1. Respuesta esperada

```
{
  "archivoPublico": "https://..."
}
```

## 6. Infraestructura

El servicio debe alojarse preferentemente en el proyecto de Firebase.

## 7. Validaciones y Manejo de Errores

### 7.1. Validaciones

- `linkArchivo` debe ser una URL válida.
- Validar que el archivo remoto exista y sea descargable.
- Validar que el tipo de archivo descargado sea PDF, JPG o JPEG.
- Validar que `tipoDeArchivo` pertenezca al conjunto permitido.

### 7.2. Errores a manejar

- Error al descargar archivo.
- Error al optimizar archivo.
- Error al subir archivo a Supabase.
- Error al generar o recuperar el enlace público.

La respuesta de error deberá contener:

```
{
  "error": "descripcion del error"
}
```

## 8. Flujo Completo del Servicio

1. Recibir solicitud con `linkArchivo`, `tipoDeArchivo` y `signedUrl`.
2. Descargar archivo remoto.
3. Validar tipo MIME.
4. Optimizar archivo según tipo.
5. Subir archivo mediante la `signedUrl` recibida.
6. Construir `public_url` según la url base de Supabase y `tipoDeArchivo`.
7. Responder con la URL pública.

## 9. Requisitos No Funcionales. Requisitos No Funcionales

- Tiempo de respuesta óptimo aun para archivos pesados.
- Uso eficiente de recursos en Cloud Functions.
- Fiabilidad en la subida de archivos.
- Logging básico para trazabilidad.