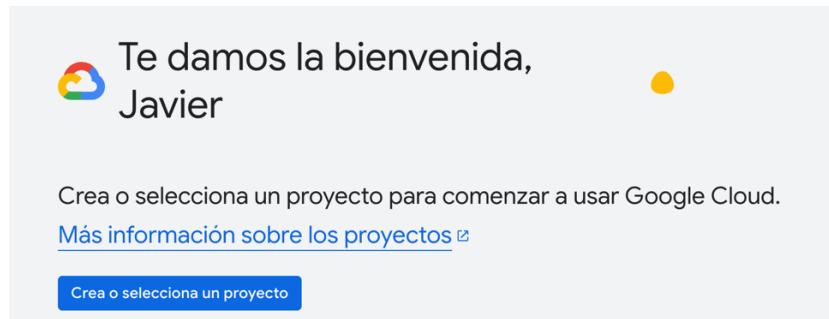


# DIA 1. CONFIGURACION AVANZADA DE LA INFRAESTRUCTURA DE GCP.

## 1. Creación y configuracion del Proyecto:

Dirigirse a la URL console.cloud.google.com/.

Para crear un proyecto seleccionar el boton Crear o selecciona un proyecto.



Esto muestra una pestaña en la cual es posible visualizar los proyectos existentes o crear uno nuevo, para crear el nuevo proyecto dar click en proyecto nuevo, siendo el boton mostrado en la esquina superior izquierda.

A screenshot of the Google Cloud project selection interface. It shows a search bar with "Buscar proyectos y carpetas" and a magnifying glass icon. Below the search bar are three tabs: "Recientes", "Destacados", and "Todos", with "Todos" being the active tab. A table lists one project: "Sin organización" (Organization type) with ID "0". In the top right corner, there is a blue button with a gear icon labeled "Proyecto nuevo".

Al seleccionar ese boton se mostraran las siguientes opciones donde se elegira el nombre del proyecto, al hacerlo y dar clic en el boton de crear proyecto mostrara la siguiente notificacion indicando el tiempo de espera de creacion del proyecto

A screenshot of a notification message. The title is "Notificaciones". It shows a green checkmark next to the text "Crear proyecto: storage-events-lifecycle" and "Hace unos instantes". Below this is a blue link labeled "Seleccionar proyecto".

Ahora seleccionar el menu de navegacion, APIs y servicios, APIs y servicios habilitados; al hacer esto mostrara los proyectos que se tienen disponibles.

The screenshot shows the Google Cloud navigation bar on the left with options like Cloud Hub, Vista general de Cloud, Soluciones, Productos fijados, and API. The API section is expanded, showing sub-options: APIs y servicios, Facturación, IAM y administración, and Marketplace. A dropdown menu for 'APIs y servicios' is open, listing: APIs y servicios habilitados, Biblioteca, Credenciales, Pantalla de consentimiento de OAuth, and Acuerdos de uso de páginas. To the right, a page titled 'APIs y servicios habilitados' displays a message: 'Para ver esta página, selecciona un proyecto.' Below it, a section titled 'Selecciona un proyecto reciente' shows a project named 'storage-events-lifecycle' with details: ID del proyecto: storage-events-lifecycle, Organización: No organization, and Último acceso: hace unos instantes.

Seleccionar el proyecto STORAGE-EVENTS-LIFECYCLE para ahora poder habilitar las APIs: Cloud Storage, Cloud Functions, Cloud Pub/Sub, Cloud Logging y Cloud Build. Seleccionar cloud shell (opcion superior izquierda de la pagina) para introducir lo siguiente. Seguido de eso, mostrara un mensaje de “Finalizado exitosamente” con la linea de la operacion

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud services enable storage.googleapis.com \
cloudfunctions.googleapis.com \
pubsub.googleapis.com \
logging.googleapis.com \
cloudbuild.googleapis.com
Operation "operations/acf.p2-196136658496-0ceb1ca5-90af-4724-88e0-79859c8d49ec" finished successfully.
tutosvorago@cloudshell:~ (storage-events-lifecycle)$
```

Dentro del mismo cloud shell se creara el usuario con permisos minimos realizando lo siguiente:

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud iam service-accounts create usuario-pruebas \
> --display-name="Usuario de pruebas"
```

Con el usuario ya creado se procede a añadir sus respectivos permisos de la siguiente manera:

Utilizar el siguiente comando para crear una variable donde se guardara el ID del proyecto

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ PROJECT_ID=$(gcloud config get-value project)
Your active configuration is: [cloudshell-22157]
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ echo $PROJECT_ID
storage-events-lifecycle
```

Escribir el siguiente comando para dar los permisos de subir archivos al bucket

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud projects add-iam-policy-binding
$PROJECT_ID \
> --member="serviceAccount:usuario-pruebas@$PROJECT_ID.iam.gserviceaccount.com" \
> --role="roles/storage.objectCreator"
Updated IAM policy for project [storage-events-lifecycle].
bindings:
```

Escribir el siguiente comando para dar el permiso de leer metadatos dentro del bucket

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud projects add-iam-policy-binding
$PROJECT_ID --member="serviceAccount:usuario-pruebas@$PROJECT_ID.iam.gserviceaccount.com"
--role="roles/storage.objectViewer"
Updated IAM policy for project [storage-events-lifecycle].
```

Escribir el siguiente comando para dar el permiso de las funciones sean ejecutadas.

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud projects add-iam-policy-binding
$PROJECT_ID --member="serviceAccount:usuario-pruebas@$PROJECT_ID.iam.gserviceaccount.com"
--role="roles/cloudfunctions.invoker"
Updated IAM policy for project [storage-events-lifecycle].
```

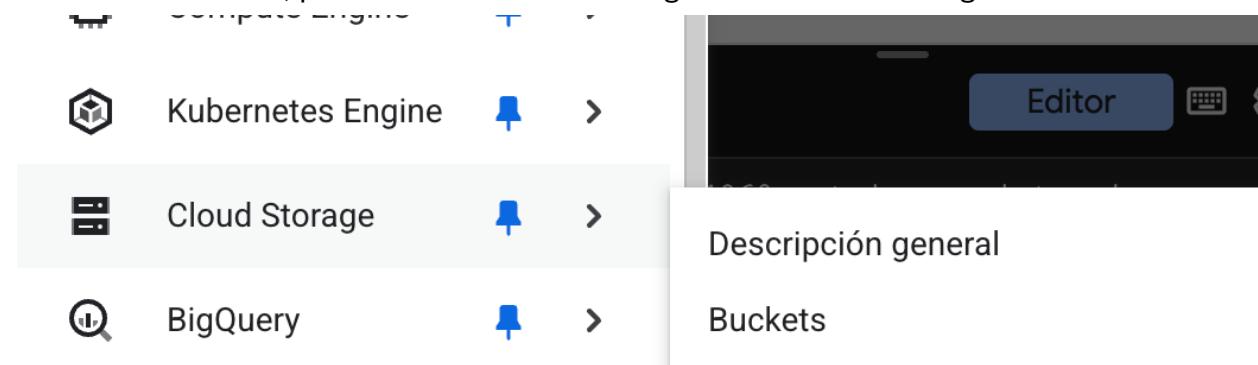
Escribir el siguiente comando para dar el permiso de que la Cloud Function escriba logs.

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud projects add-iam-policy-binding
$PROJECT_ID --member="serviceAccount:usuario-pruebas@$PROJECT_ID.iam.gserviceaccount.com"
--role="roles/logging.logWriter"
Updated IAM policy for project [storage-events-lifecycle].
```

## 2. Diseño de Almacenamiento y Automatización Inicial:

### Crear un bucket en cloud storage.

Se creara un bucket, para ello ir a Menu de navegacion -> Cloud Storage -> Buckets



Dar click en crear buckets.

Buckets

+ Crear

Ahora introducir el nombre, en este caso INPUT-FILES-PRUEBA

- Comenzar

Selecciona un nombre permanente globalmente único. [Lineamientos para asignar nombre](#)

input-files-prueba

Sugerencia: No incluyas información sensible

Seleccionar la region US-CENTRAL1 del bucket

Region

Latencia mínima dentro de una sola región

us-central1 (Iowa)

Seleccionar la manera de almacenar en estandar

- **Elige cómo almacenar tus datos**

Una clase de almacenamiento determina los costos del almacenamiento, la recuperación y las operaciones con diferencias mínimas en el tiempo de actividad. Elige si deseas administrar los objetos de forma automática o especifica una clase de almacenamiento predeterminada según la duración con la que planeas almacenar tus datos y tu carga de trabajo o caso de uso.

[Learn more](#)

Autoclass [?](#)

Realiza la transición automáticamente de cada objeto a las clases Estándar o Nearline según la actividad a nivel de objeto para optimizar el costo y la latencia. Se recomienda si la frecuencia de uso puede ser impredecible. Se puede cambiar a una clase predeterminada en cualquier momento. [Detalles de precios](#)

Configurar una clase predeterminada

Se aplica a todos los objetos del bucket, a menos que modifiques de forma manual la clase por objeto o establezcas reglas de ciclo de vida de los objetos. Se recomienda cuando el uso es muy predecible.

Standard

La mejor opción para el almacenamiento a corto plazo y los datos de acceso frecuente

Dejar la configuracion como muestra en la opcion de como controlar el acceso a los objetos

- Elige cómo controlar el acceso a los objetos

#### Impedir el acceso público

Restringe el acceso público a los datos a través de Internet Esto evitará que el bucket se use para el hosting web. [Más información](#)

Aplicar la prevención de acceso público a este bucket

#### Control de acceso

##### Uniforme

Garantiza el acceso uniforme a todos los objetos del bucket mediante el uso exclusivo de permisos a nivel de bucket (IAM). Esta opción se aplicará de manera permanente después de 90 días. [Más información](#)

##### Preciso

Especifica el acceso a objetos individuales mediante el uso de permisos a nivel de objeto (LCA) además de los permisos a nivel de bucket (IAM). [Más información](#)

[Continuar](#)

Y ahora desmarcar la POLITICA DE ELIMINACION NO DEFINITIVA

- Elige cómo proteger los datos de objeto

Tus datos siempre están protegidos con Cloud Storage, pero también puedes elegir entre estas opciones adicionales de protección de datos para agregar capas de seguridad extras.

#### Protección de datos

##### Política de eliminación no definitiva (para la recuperación de datos)

Cuando se habilita esta opción, el bucket y sus objetos se conservan durante un período específico después de que se borran, y pueden restablecerse durante este tiempo. [Más información](#)

##### Control de versiones de objetos (para el control de versión)

Para restablecer objetos borrados o reemplazados. Para minimizar el costo de almacenamiento de versiones, te recomendamos limitar la cantidad de versiones no actuales por objeto y programarlas para que venzan después de una cantidad de días. [Más información](#)

##### Retención (para el cumplimiento)

Para impedir que se borrar o modifiquen los objetos del bucket durante un período específico.

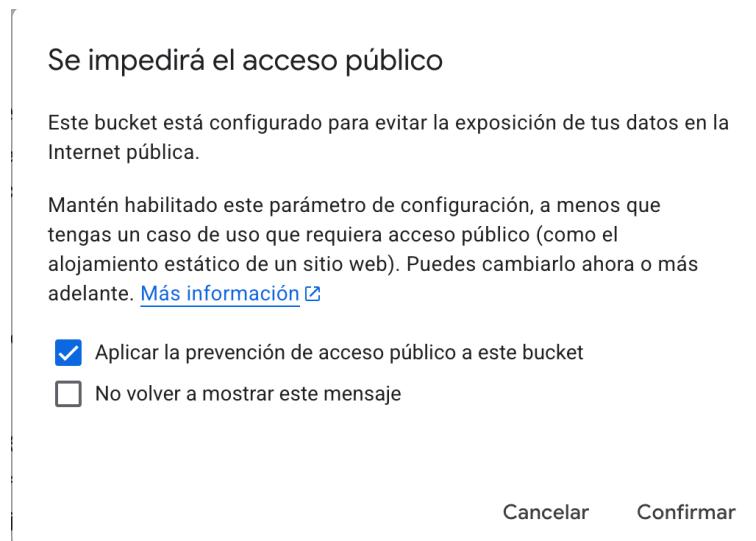
Y por ultimo dar click en el boton Crear

## ✓ Encriptación de datos

[Crear](#)

[Cancelar](#)

Al seleccionar el botón crear se mostrara la siguiente ventana, sin necesidad de seleccionar algo mas, dar click en confirmar.



En la interfaz ahí mismo en el bucket, seleccionar Ciclo de vida (Lifecycle) -> Agregar regla (Add rule) -> Borrar objeto (Delete) -> Edad (Age) = 30

Objetos      Configuración      Permisos      Protección      **Ciclo de vida**

**i** Luego de agregar o editar una regla, los cambios pueden tardar hasta 24 horas en aplicarse.

Las reglas de ciclo de vida te permiten aplicar acciones a los objetos de un bucket cuando se cumplen ciertas condiciones; por ejemplo, cambiar objetos a clases de mayor almacenamiento en frío cuando alcanzan o superan una antigüedad determinada. [Más información](#)

Si un objeto satisface las condiciones de múltiples reglas, ocurre lo siguiente:

- La eliminación tiene mayor prioridad que un cambio de clase de almacenamiento.
- Se prioriza el cambio de objetos a las clases de almacenamiento más frías en lugar de las más cálidas (p. ej., los objetos se cambiarán a la clase de almacenamiento Archive en lugar de Coldline si hay reglas para ambas).

Reglas      [Agregar una regla](#)      Borrar todo

Al seleccionar agregar una regla, se mostrara lo siguiente, para siguiente seleccionar la opcion borrar objeto

- **Selecciona una acción**

- Establecer la clase de almacenamiento en Nearline  
Ideal para copias de seguridad y datos a los que se accede menos de una vez al mes
- Establecer la clase de almacenamiento en Coldline  
Ideal para recuperación ante desastres y datos a los que se accede menos de una vez por trimestre
- Establecer la clase de almacenamiento en Archive  
La mejor opción para la conservación digital a largo plazo de los datos a los que se accede menos de una vez al año
- Borrar objeto

En las condiciones del objeto que sera la opcion siguiente a seleccionar, marcar la casilla antigüedad y añadir 30

#### Establecer condiciones

Antigüedad [?](#)

Ingresar la edad \* —————

30	días
----	------

La antigüedad se cuenta a partir del momento en que un objeto se subió a su ubicación actual.

Seleccionar el boton de crear, basicamente borrara archivos despues de 30 días.

- Selecciona una acción**

Acción: Borrar objeto

- Selecciona las condiciones del objeto**

Antigüedad:30

**Crear**

Cancelar

Ahora, despues de crear la regla, seleccionar la opcion Permisos en los detalles de bucket con el objetivo de aplicar minimo privilegio al bucket.

Objetos      Configuración      **Permisos**      Protección

### Acceso público

### No público

Marcar al usuario de pruebas que aparece en la sección de permisos

<input checked="" type="checkbox"/> 	usuario-pruebas@storage-events-lifecycle.iam.gserviceaccount.com	Usuario de pruebas	Creador de objetos de Storage Visualizador de objetos de Storage
---	--	--------------------	---

Seleccionar la opción otorgar acceso

 **Otorgar acceso**

Seleccionar el service account

Agregar principales

Las principales son usuarios, grupos, dominios o cuentas de servicio. [Más información sobre las principales de IAM](#)

Principales nuevas \*

X ?

Ahora sigue el momento de asignar los roles siendo 2 los que se buscan, storage object creator y storage object viewer

Seleccionar Cloud Storage -> Creador de objetos de Storage

Cloud Storage	almacenamiento
Por producto o servicio	Administrador de objetos de Storage
Administración	<b>Creador de objetos de Storage</b>

Despues de hacer eso, se mostrara lo siguiente, solo sera necesario dar click en guardar

Agregar principales

Las principales son usuarios, grupos, dominios o cuentas de servicio. [Más información sobre las principales de IAM](#)

Principales nuevas \*

usuario-pruebas@storage-events-lifecycle.iam.gserviceaccount.com

Asignar roles

Los roles se componen de conjuntos de permisos y determinan lo que la principal puede hacer con este recurso. [Más información](#)

Rol \* Creador de objetos de Storage

Permite a los usuarios crear objetos.  
No permite ver, borrar ni reemplazar objetos.

Condición de IAM (opcional) [?](#)

+ Agregar condición de IAM

+ Agregar otro rol

[Guardar](#) [Cancelar](#)

Nuevamente otorgamos al mismo usuario acceso para agregar otro rol

[Ver por principales](#)

[+ Otorgar acceso](#)

Seleccionar el mismo principal de la anterior con el rol Cloud Storage -> Visualizador de objetos de Storage y click en guardar.

### Agregar principales

Las principales son usuarios, grupos, dominios o cuentas de servicio. [Más información sobre las principales de IAM](#)

Principales nuevas \*

usuario-pruebas@storage-events-lifecycle.iam.gserviceaccount.com

### Asignar roles

Los roles se componen de conjuntos de permisos y determinan lo que la principal puede hacer con este recurso. [Más información](#)

Rol \* Visualizador de objetos de St...

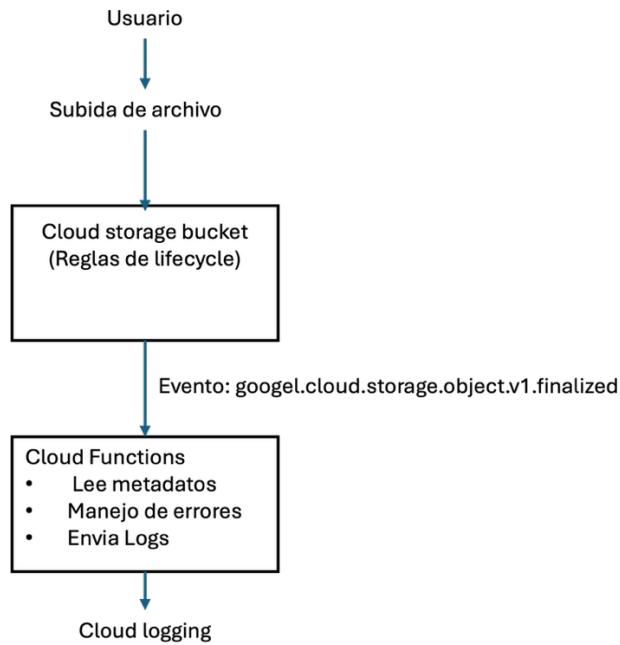
Otorga acceso para ver los objetos y sus metadatos, excepto las LCA. También puede enumerar los objetos de un bucket.

Condición de IAM (opcional) [?](#)

+ Agregar condición de IAM

[Guardar](#) [Cancelar](#)

## DIAGRAMA REPRESENTATIVO DEL FLUJO COMPLETO



**Desarrollar una Cloud Function (en node.js) que se active automáticamente al subir un archivo al bucket.**

Para esto, crear un archivo index.js en el Cloud Shell

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ nano index.js
```

En este archivo se ingreso el siguiente código

```
exports.onFileUpload = async (event, context) => {
  try {
    // El parámetro "event" contiene los metadatos del objeto que activó la función.
    const file = event;

    // Extraer metadatos principales del archivo subido.
    // En caso de que algún valor no exista, se asigna un valor por defecto.
    const name = file.name || "desconocido";           // Nombre del archivo
    const size = file.size || 0;                         // Tamaño en bytes
    const contentType = file.contentType || "sin-tipo"; // Tipo MIME del archivo

    // Registrar los metadatos en Cloud Logging para auditoría y depuración.
    console.log(`
      Archivo recibido:
      - Nombre: ${name}
      - Tamaño: ${size} bytes
      - Tipo: ${contentType}
    `);

    // La función retorna un objeto para confirmar su ejecución correcta.
    return {
      status: "OK",
      fileName: name,
    };
  } catch (error) {
    // En caso de error, se registra un mensaje en Cloud Logging.
    console.error("Error procesando archivo:", error);

    // Se lanza una excepción para indicar que la ejecución falló.
    throw new Error("Cloud Function failed.");
  }
};
```

Lo siguiente es, en el mismo Cloud Shell se crea un archivo package.json

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ nano package.son
```

E ingresar lo siguiente

```
package.json > ...
1  {
2    "name": "storage-file-processor",
3    "version": "1.0.0",
4    "main": "index.js",
5    "dependencies": {}
6  }
7
```

Ahora se procese a realizar el despliegue la función

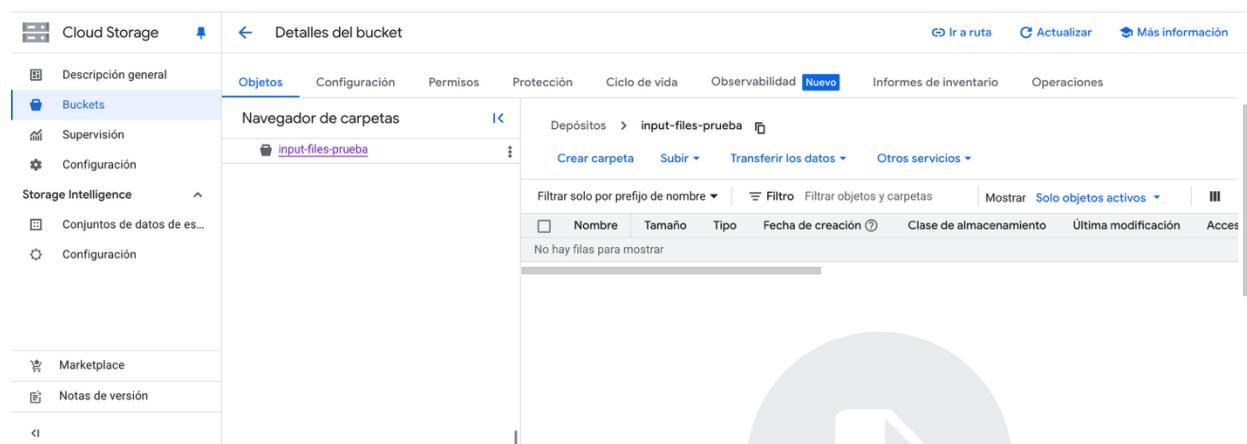
```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ gcloud functions deploy onFileUpload \
--no-gen2 \
--runtime=nodejs20 \
--trigger-resource=input-files-prueba \
--trigger-event=google.storage.object.finalize \
--entry-point=onFileUpload \
--region=us-central1

Deploying function (may take a while - up to 2 minutes)...working
```

Para saber si se desplego correctamente solo es mirar el status

```
sourceUploadUrl: https://storage.googleapis.com/uploads.appspot.com/5c31b4bd-a1cd-4f3d-b7f0-e2e27b556489
status: ACTIVE
timeout: 60s
updateTime: '2025-11-29T22:07:43.880761854Z'
versionId: '1'
tutosvorago@cloudshell:~ (storage-events-lifecycle)$
```

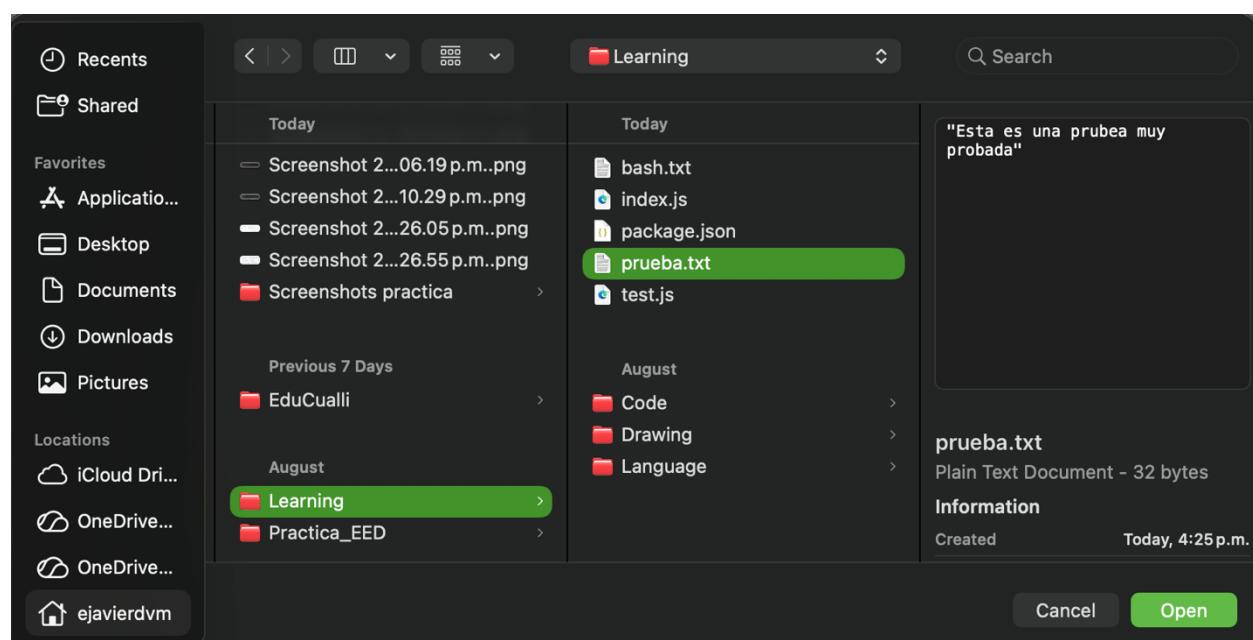
Para probar la función, sera subir un archivo llamado prueba.txt siendo dentro del bucket



Seleccionar la opcion subir -> subir archivos

The screenshot shows the Google Cloud Storage interface for a bucket named "input-files-prueba". At the top, there are buttons for "Crear carpeta" (Create folder), "Subir" (Upload), and "Transferir" (Transfer). A dropdown menu is open over the "Subir" button, with the option "Subir archivos" (Upload files) highlighted. Below this, there is a search bar labeled "Filtrar solo por prefijo" (Filter only by prefix) and a "Fil" (File) icon.

Seleccionar el archivo a utilizar en la computadora, en este caso prueba.txt



El archivo seleccionado ahora esta dentro del bucket

Nombre	Tamaño	Tipo	Fecha de creación	Clase de almacenamiento
prueba.txt	32 B	text/plain	29 nov 2025, 4:28:38 p.m.	Standard

Ahora en el explorador, buscar lo siguiente “Explorador de registros”

The screenshot shows the Google Cloud Logging interface. At the top, it says "Explorador de registros" and has a "Buscar" (Search) button. Below this, there is a section titled "Resultados principales" (Main results) which includes a link to "Explorador de registros Página del producto · Logging". The bottom navigation bar includes "Registros del proyecto" and "resource.type".

Ya dentro del explorador de registros, en la barra de busqueda agregar

y despues dar click en el boton Ejecutar consulta

para seguido mostrar los siguientes datos, donde indica el archivo, su peso y la informacion que se requeria.

The screenshot shows the Google Cloud Platform Log Explorer interface. At the top, there is a search bar with the query: "resource.type='cloud\_function' resource.labels.function\_name='onFileUpload'". Below the search bar, there are several filter and sorting options. On the right side, there is a blue button labeled "Ejecutar consulta" (Execute query) and a checked checkbox labeled "Mostrar consulta" (Show query). The main area displays a timeline from 15:30:00 to 17:04:00. A horizontal bar at the bottom indicates the execution time for each log entry. The results table has columns: GRAVEDAD, HORA, and RESUMEN. The first few rows show log entries for file uploads, including details like the file name, size, type, and status ('ok').

GRAVEDAD	HORA	RESUMEN
> *	2025-11-29 17:01:08.475	onFileUpload xp0yxc6buhms FUNCTION EXECUTION TOOK 72 MS, FINISHED WITH STATUS: OK
> *	2025-11-29 17:01:08.475	onFileUpload xp0yxc6buhms Function execution started
> *	2025-11-29 17:01:08.478	onFileUpload xp0yxc6buhms Archivo procesado: Nombre: Screenshot 2025-11-29 at 2.09.24 p.m..png Tamaño: 65352 bytes Tipo: image/png
> *	2025-11-29 17:01:08.479	onFileUpload xp0yxc6buhms Function execution took 3 ms, finished with status: 'ok'
> *	2025-11-29 17:02:41.375	onFileUpload xp0ygnf2flz6 Function execution started
> *	2025-11-29 17:02:41.380	onFileUpload xp0ygnf2flz6 Archivo procesado: Nombre: prueba.txt Tamaño: 32 bytes Tipo: text/plain
> *	2025-11-29 17:02:41.382	onFileUpload xp0ygnf2flz6 Function execution took 6 ms, finished with status: 'ok'

Aquí con mas detalle el archivo de prueba.txt que se subio.

```
> * 2025-11-29 17:02:41.380 (onFileUpload) xp0ygnf2flz6 Archivo procesado: Nombre: prueba.txt Tamaño: 32 bytes Tipo: text/plain
```

### 3. Pruebas y documentacion

Implementar pruebas basicas unitarias para la Cloud Function.

Instalar Jest como dependencia usando NPM –save-dev jest

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ npm install --save-dev jest
```

Abrir el package.json que se crearon en los pasos anteriores.

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ nano package.json
```

Y colocar lo siguiente dentro del archivo

```
GNU nano 7.2
{
  "name": "storage-file-processor",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "jest"
  },
  "devDependencies": {
    "jest": "^29.0.0"
  }
}

```



Crear una carpeta para las pruebas

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ mkdir tests
```

Ahora crear el archivo dentro de la carpeta tests

```
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ nano tests/onFileUpload.test.js
```

agregar lo siguiente dentro del archivo. Lo que se busca es tener un flujo correcto y un flujo de error bien manejado y logueado

```
5   // Grupo de pruebas para la función "onFileUpload".
6   describe("Cloud Function onFileUpload", () => {
7
8     // Caso de prueba: Validar que la función procese correctamente
9     // un archivo válido simulando el evento enviado por Cloud Storage.
10    test("procesa correctamente un archivo válido", async () => {
11
12      // Evento falso (mock) que simula los metadatos de un archivo subido.
13      const fakeEvent = {
14        name: "archivo-prueba.txt",
15        size: 1500,
16        contentType: "text/plain",
17      };
18      const result: any
19      // Eje const result = await onFileUpload(fakeEvent, {});
20      const result = await onFileUpload(fakeEvent, {});
21
22      // Validamos que la función retorne un estado OK.
23      expect(result.status).toBe("OK");
24
25      // Validamos que el nombre devuelto coincida con el archivo simulado.
26      expect(result.fileName).toBe("archivo-prueba.txt");
27    });
28
29    // Caso de prueba: Validar el manejo de errores.
30    // Si el evento recibido es nulo, la función debe lanzar una excepción.
31    test("lanza un error cuando el evento es nulo", async () => {
32
33      // Se espera que la función falte al recibir un evento inválido.
34      await expect(onFileUpload(null, {})).rejects.toThrow();
35    });
36  });
}
```

En el cloud shell se veria asi

```
GNU nano 7.2                               tests/onFileUpload.test.js *
// Importamos la función desde index.js
const { onFileUpload } = require("../index");

describe("Cloud Function onFileUpload", () => {
  test("procesa correctamente un archivo válido", async () => {
    const fakeEvent = {
      name: "archivo-prueba.txt",
      size: 1500,
      contentType: "text/plain",
    };

    const result = await onFileUpload(fakeEvent, {});

    expect(result.status).toBe("OK");
    expect(result.fileName).toBe("archivo-prueba.txt");
  });

  test("lanza un error cuando el evento es nulo", async () => {
    await expect(onFileUpload(null, {})).rejects.toThrow();
  });
});
```

Entonces, ahora para poder analizar si funciono o no todo esto, es momento de ejecutar las pruebas escribiendo npm test en la raiz donde se tiene package.json mostrando el siguiente resultado.

```
tutosvorago@cloudshell:~/storage-events-lifecycle$ npm test
> storage-file-processor@1.0.0 test
> jest

  console.log
    Archivo procesado: Nombre: archivo-prueba.txt Tamaño: 1500 bytes Tipo: text/plain
      at log (index.js:8:13)

  console.error
    Error rprocesando archivo: TypeError: Cannot read properties of null (reading 'name')
      at name (/home/tutosvorago/index.js:5:23)
      at Object.onFileUpload (/home/tutosvorago/tests/onFileUpload.test.js:19:18)
      at Promise.finally.completed (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:1557:28)
      at new Promise (<anonymous>)
      at callAsyncCircusFn (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:1497:10)
      at _callCircusTest (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:1007:40)
      at processTicksAndRejections (node:internal/process/task_queues:103:5)
      at _runTest (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:947:3)
      at /home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:849:7
      at _runTestsForDescribeBlock (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:862:11)
      at _runTestsForDescribeBlock (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:857:11)
      at run (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:761:3)
      at runAndTransformResultsToJestFormat (/home/tutosvorago/node_modules/jest-circus/build/jestAdapterInit.js:1918:21)
      at jestAdapter (/home/tutosvorago/node_modules/jest-circus/build/runner.js:101:19)
      at runTestInternal (/home/tutosvorago/node_modules/jest-runner/build/index.js:275:16)
      at runTest (/home/tutosvorago/node_modules/jest-runner/build/index.js:343:7)

13 |
14 |   } catch (error) {
> 15 |     console.error("Error rprocesando archivo: ", error);

```

```
16 |         throw new Error("Cloud Function Failed.");
17 |
18 |     }
19 |
20 |     at error (index.js:15:13)
21 |     at Object.onFileUpload (tests/onFileUpload.test.js:19:18)

PASS tests/onFileUpload.test.js
  Cloud Function onFileUpload
    ✓ procesa correctamente un archivo válido (23 ms)
    ✓ lanza un error cuando el evento es nulo (13 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
S_snapshots: 0 total
Time:        0.437 s
Ran all test suites.
tutosvorago@cloudshell:~ (storage-events-lifecycle)$ █
```