|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre:** | CARLOS MAURICIO OTALVARO OSPINA | Fecha: | 11/12/2024 |

1. ¿Puedes explicar la diferencia entre aplicaciones de lienzo y aplicaciones basadas en modelo en Power Apps?. Mencionar pros y contras de cada enfoque y ejemplo concretos de cuando usar cada opción.

**Aplicaciones de Lienzo o Canvas :** En Power Apps, este tipo de aplicaciones permite total libertad para personalizar la interfaz, tienes la posibilidad de crear formularios desde Cero.

Aquí decides qué mostrar y cómo hacerlo (botones, campos, imágenes, dropdowns entre otros y conectarse con diferentes tipos de fuentes de datos.

Pros**:** Flexibilidad “absoluta” en el diseño, con algunos pequeños impedimentos en cuanto a lectura de datos, ideal para experiencias de usuarios que conocen poco de desarrollo de software.

Contras**:** Requiere más tiempo en diseño y puede ser más complejo integrarlas con otras plataformas. Muy sesgado a lo que permite desde la forma básica de creación,

* La plataforma está limitada para modificar el comportamiento de ciertos controles nativos. Si se requiere una funcionalidad específica que no se puede lograr con las opciones predeterminadas, es necesario crear un componente personalizado utilizando JavaScript.
* Si se necesita acceder a una fuente de datos externa que no cuenta con un conector oficial, se deben realizar llamados directos a las APIs. En estos casos, la seguridad dependerá de las capas implementadas en dichas APIs.
* UN punto super importante en el que se falla mucho es en el tema de la delegación, no todas las funciones de búsqueda disponibles en Power Apps son delegables a todas las fuentes de datos. Si un desarrollador no toma en cuenta esta limitación al crear consultas, podría enfrentarse a una pérdida de datos cuando trabaje con tablas que superen los 500 o 2000 registros, dependiendo del valor configurado como límite máximo.
* Al realizar consultas en galerías, inicialmente no se muestran todos los registros de la base de datos. Power Apps carga de forma progresiva: primero trae los primeros 200 registros y luego, a medida que el usuario hace scroll, carga bloques adicionales de 100 registros. Esto debe considerarse al diseñar aplicaciones que trabajen con grandes volúmenes de datos.

En mi caso, tuve l aoportunidad de desarrollar una aplicación desde cero llamada SIA mientras estuve aprendiendo Power Apps de tipo Canvas, esta permite hoy almacenar la información de datos de todas las implementaciones de MAC que existen hoy en día.Utiliza, SQL Server, sharepoint, procedimientos almacenados en SQL Server y se conecta con otras aplicaciones en la organización. De igual forma apoyé e desarrollo de una más robusta llamada Gobernexa que permite manejar asambleas y juntas en la organización, manejada en un 80% con componentes reutilizables.

**Aplicaciones Basadas en Modelo o Model Driven Apps**: En estas todo gira en torno a un modelo de datos centralizado, como Dataverse. La interfaz es estandarizada y ante todo se basa en la gestión de procesos.

Pros**:** Menor tiempo de desarrollo al reutilizar componentes estándar, ideal para flujos estructurados.

Contras**:** Poco personalizable visualmente.

Si se tiene un modelo de datos en una base de datos definido claramente una model driven es al opción porque muy rápidamente permite entregar la aplicación “casi lista” con algunos cambios pero con las asociaciones claras en la base de datos sin requerir mucho cambio.

1. ¿Cómo manejarías la autenticación y autorización en una aplicación de Power Platform?

Para mi depende mucho del presupuesto que se tenga, y el tiempo, por ejemplo se puede crear una en la que no se requieran conectores especiales y asignar permisos desde la plataforma a unos usuarios especiales sin que requieran licencias de power apps.

Ahora una solución más robusta y económica podría ser, asignar permisos mediante Azure Active Directory AAD, creando un modelo de roles y grupos desde base de datos, y creando un crud para agregar, modificar o bloquear usuarios de acuerdo a lo que quiera, asignaría una bolsa de licencias para que no se haga costoso e identificaría cuantos usuarios entran al mismo tiempo para disminuir costos.

Un ejemplo claro dentro de la organización, desarrollado en una aplicación críticas como Gobernexa, utiliza Azure Active Directory (AAD) para gestionar roles y accesos. Configuración de grupos de seguridad para que solo perfiles autorizados, como gerentes, así solo tienen acceso a funcionalidades específicas, como aprobar reuniones o registrar datos sensibles. Esto asegura una capa de seguridad sólida, sin complicar la experiencia de usuario.

Para resumir, AAD más un modelo de base de datos que permita dar permisos por cada una de las funciuonalidades de la aplicación teniendo en cuenta roles y grupos.

1. ¿Qué estrategias utilizarías para optimizar el rendimiento de un flujo que procesa grandes volúmenes de datos en Power Automate?
2. Filtrar datos en el origen utilizando los conectores evitando traer datos innecesarios. Por ejemplo un flujo que consulta txs masivas, filtro por fechas relevantes desde base de datos.
3. Configurar límites en las acciones para dividir los datos en bloques manejables (paginación).
4. Seleccionar la opción de paralelismo dentro del flujo para poder ejecutar varios hilos a la vez.
5. ¿Qué son los Custom Connectors y los Custom Actions en Power Automate? Explique casos de usos donde se podrían implementar, pros y contras.

Los Custom Connectors, los uso cuando necesito conectar APIs externas no soportadas de manera nativa.

Ejemplo: Para integrar datos de tráfico en tiempo real de Google Maps en un flujo, diseñé un conector personalizado que consulta su API oficial.

Los Custom Actions son perfectos para procesos específicos utilizando Dataverse. Un custom action es muy parecido a un procedimiento almacenado de base de datos, en el Custom Action se centraliza la lógica de negocio en Dataverse mientras en el SP se centraliza la lógica de negocio dentro de la Base de Datos. Ambas permiten que s reutilice la lógica sin necesidad de duplicarla y ser llamada en flujos o aplicaciones. Tienen entradas y salidas. La única diferencia es que el Custom Action es más felxible pues permite integraciones con otras herramientas mientras el SP solo en el entorno de base de datos. Adicional, el Custom Action no requiere conexiones directas pues hace parte de la plataforma y utiliza conectores nativos, en un SP si se requeriría una conexión directa a la BD.

1. ¿Cómo implementarías un manejo de errores robusto en un flujo de Power Automate?

No se qué tan detallado se requiera en la respuesta pero yo haría lo siguiente:

1. Reintentar: Usa políticas de reintentos automáticos en las acciones críticas y valida que los datos de entrada estén completos y en el correctamente formateados.
2. Detectar: Usa la configuración “Run After” para ejecutar pasos específicos solo si una acción falla. Agrupa acciones en “Scopes” para encapsular y manejar errores dentro de bloques.
3. Registrar: Definir un histórico centralizado de errores y tipos de errores(log). Puede ser enSharepoint o Dataverse o una base de datos transaccional, ojalá con Colas. Donde esté el nomrbe del flujo, hora, error, tipo de error y alcance.
4. Notificar: Informar a las personas el fallo. Envíar un mensaje, alerta o indicación de donde ocurrió el error en el flujo y de qué tipo.
5. Corregir: Así mismo utilizas Scopes para manejar errores como en los try-catch para que se informe y el flujo pueda continuar con el siguiente registro sin que se detenga totalmente. Y en algunos casos usar un finally para limpiar lo que se tenga que limpiar por no finalizar correctamente por un error.
6. ¿Qué técnicas utilizarías para reintentar automáticamente acciones fallidas en un flujo?

Configuraría un Retry Policy para reintentar automáticamente en acciones críticas si hay fallos: Ultilizando un número máximo de intentos, definiendo un intervalo entre intentos yasea fijo o exponencial. Y para informar utilizaría variables que permitan rastrear el estado del flujo si se agotan los reintentos se enviaría una notificación. Un ejemplo sería si tengo un flujo que consulta datos de una API externa, en caso de que falla por un error de red, el Retry Policy intenta nuevamente hasta X veces, aumentando el intervalo entre cada intento (por ejemplo: 5, 10, 20 y 40 segundos).

1. Imagina que has realizado un análisis de datos y has encontrado información valiosa. ¿Cómo presentarías tus hallazgos a un equipo no técnico?

Lo explicaría de forma clara y visual para que todos lo entiendan, algo como lo siguiente:

Ir al Grano: Resumo los datos más importantes que tienen impacto directo en las decisiones del equipo. Esto incluye métricas destacadas y cualquier insight que pueda influir en las estrategias.

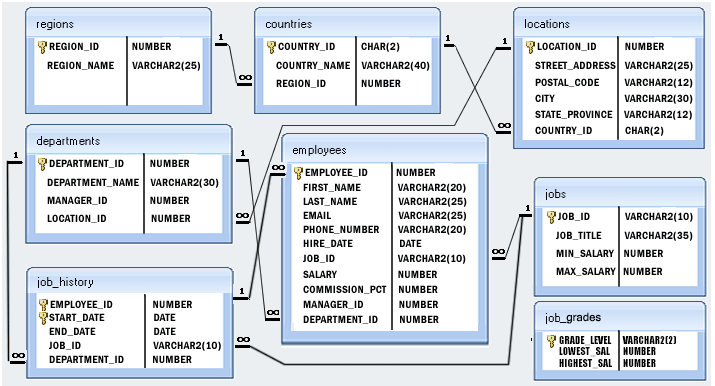
Visualizar con Power BI: Creo gráficos simples, como barras o líneas, que muestren lo relevante. Por ejemplo, en un análisis de transacciones fraudulentas, mostrando un gráfico que indique las horas y montos más propensos a actividades sospechosas.

Contar una historia (storytelling): Hago que los datos se conecten con algo real. Por ejemplo, si el análisis identifica un aumento de fraudes entre las 6:00 y 8:00 a. m., lo presentaría como “El punto ciego de las mañanas” y propondría soluciones, como mayor monitoreo en ese horario.

Dar ejemplos claros: Relaciono los datos con casos concretos para que vean el impacto directo.

Hacerlo fácil de entender:  podría mostrar cómo una serie de transacciones menores a $5.000 en un período corto llevó a pérdidas acumuladas significativas.

Por ultimo crearía una visaulización con una breve explicación para garantizar que el mensaje sea claro.



1. Escribe un query SQL que retorne la cantidad de empleados que hay en cada departamento. Se requiere los departamentos que tienen más de 5 empleados. Debe retornar el nombre del departamento, el nombre de la ciudad y país del departamento y la cantidad de empleados.

SELECT

d.department\_name AS Departamento,

l.city AS Ciudad,

c.country\_name AS País,

COUNT(e.employee\_id) AS Cantidad\_Empleados

FROM departments d

JOIN

locations l ON d.location\_id = l.location\_id

JOIN

countries c ON l.country\_id = c.country\_id

JOIN

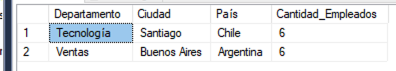
employees e ON d.department\_id = e.department\_id

GROUP BY

d.department\_name, l.city, c.country\_name

HAVING

COUNT(e.employee\_id) > 5;

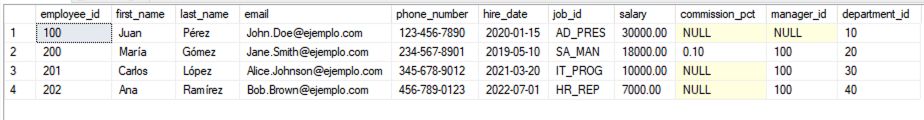


1. Escribe un query SQL para encontrar los managers. Devuelva la información completa de los managers.

SELECT DISTINCT m.\*

FROM employees e

JOIN employees m ON e.manager\_id = m.employee\_id;



1. Escribe un query SQL que calcule la experiencia de los managers. Debe retornar la siguiente información del manager: ID, Nombre, nombre del trabajo, nombre del departamento, nombre de la ciudad, nombre del país, fecha de contratación, salario y experiencia, indicando años, meses y días.

SELECT

m.employee\_id AS Manager\_ID,

CONCAT(m.first\_name, ' ', m.last\_name) AS Nombre,

j.job\_title AS Trabajo,

d.department\_name AS Departamento,

l.city AS Ciudad,

c.country\_name AS País,

m.hire\_date AS Fecha\_Contratación,

m.salary AS Salario,

DATEDIFF(YEAR, m.hire\_date, GETDATE()) AS Años\_Experiencia,

DATEDIFF(MONTH, m.hire\_date, GETDATE()) % 12 AS Meses\_Experiencia,

DATEDIFF(DAY, DATEADD(MONTH, DATEDIFF(MONTH, m.hire\_date, GETDATE()), m.hire\_date), GETDATE()) AS Días\_Experiencia

FROM

employees m

JOIN

jobs j ON m.job\_id = j.job\_id

JOIN

departments d ON m.department\_id = d.department\_id

JOIN

locations l ON d.location\_id = l.location\_id

JOIN

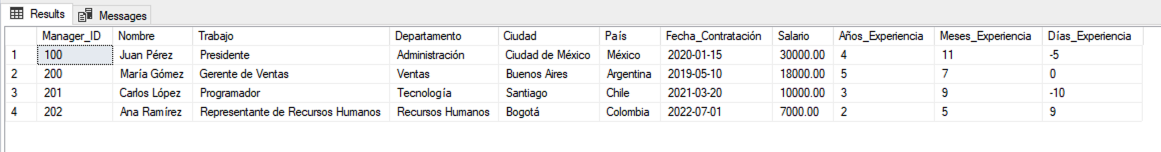
countries c ON l.country\_id = c.country\_id

WHERE

m.employee\_id IN (SELECT DISTINCT manager\_id

FROM employees WHERE manager\_id IS NOT NULL

);



**Caso 1. Análisis de datos.**



El fraude es una amenaza constante en todas las escalas comerciales. Detectarlo requiere tecnología avanzada y análisis profundo de datos para identificar patrones anómalos y salvaguardar activos. Desde el robo de identidad hasta el fraude financiero, las organizaciones deben estar vigilantes. La detección implica una combinación de tecnología, experiencia y diligencia para mantener a raya a los estafadores y proteger la integridad de la empresa. Es un proceso continuo y dinámico en la lucha contra la actividad delictiva.

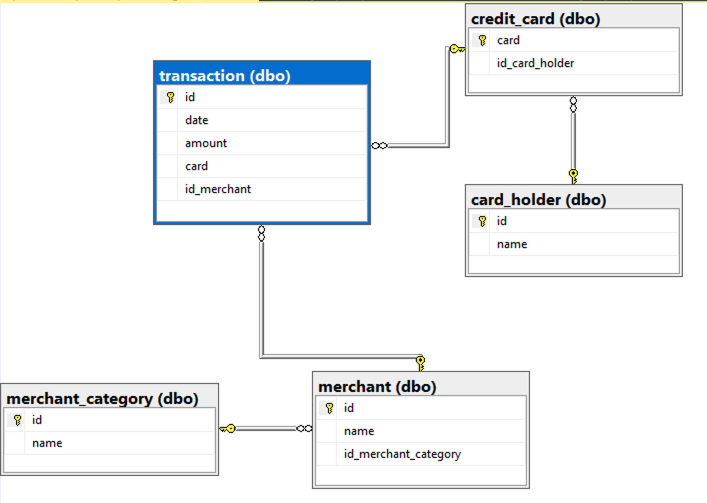
Tu misión es identificar actividades fraudulentas por medio del análisis de datos, cuentas con información de transacciones de tarjetas de crédito para realizar tu labor: [Conjunto de datos](https://drive.google.com/drive/folders/1F0nP_7Ttp-LS2HNKIlBW1f8OOHR8AVjg?usp=sharing)

El director de auditoría con el que trabajas te menciona actividades que son consideradas sospechosas:

1. Transacciones de montos altos durante las primeras horas de la mañana, es decir, entre las 6:00 a 8:00 a. m.
2. Algunos estafadores efectúan múltiples transacciones de bajo valor (generalmente menos de $2,00) utilizando tarjetas de crédito, lo cual suele pasar desapercibido para los titulares de estas.

Realiza un análisis descriptivo con los datos provisto para identificar actividades fraudulentas/anómalas.

* Crea un Diagrama Entidad Relación (DER) con base a los archivos CSV proporcionados. Usa la herramienta de diagramación de tu preferencia.



* Debes realizar esta tarea en un Notebook usando tu lenguaje de preferencia (Python, R).
* Entrega un video de no más de 15 minutos explicando los resultados y técnicas utilizadas.

https://drive.google.com/drive/folders/1GKOK0AqhMi2LyMq0Fq-l2Nov6sYRcuIc?usp=drive\_link

* Todos los entregables se deben subir a Github privado y debes compartir el acceso al repositorio.

https://github.com/mincho77/Fraude\_Transacciones

**Caso 2. Transformación Digital.**



Diseña un flujo de trabajo que automatice el proceso de aprobación de gastos. El flujo debe iniciar cuando un empleado envía un formulario de gastos, enviar una notificación al gerente para su aprobación, y actualizar una lista de SharePoint con el estado de la solicitud una vez aprobada o rechazada.

**Objetivo**: Diseñar un flujo de trabajo en Power Automate que automatice el proceso de aprobación de gastos.

**Requisitos**:

* **Inicio del Flujo**: El flujo debe iniciarse cuando un empleado envía un formulario de gastos. Este formulario debe ser una Power Apps para el registro de datos y permita adjuntar los comprobantes de los gastos, fuente de datos una lista de SharePoint o Dataverse (puede usar dataverse for teams).
* **Notificación al Gerente**: Una vez que el formulario es enviado, el flujo debe enviar una notificación por correo electrónico al gerente correspondiente para su aprobación (usar flujos de aprobación de teams o adaptive cards). El correo debe contener los detalles del gasto y un enlace/boton para aprobar o rechazar la solicitud.
* **Aprobación/Rechazo del Gasto**: El gerente debe poder aprobar o rechazar la solicitud directamente desde el correo electrónico. ***Si el gasto es aprobado***, el flujo debe actualizar la lista de SharePoint con el estado "Aprobado" y enviar una notificación al empleado informándole de la aprobación. ***Si el gasto es rechazado***, el flujo debe actualizar la lista de SharePoint con el estado "Rechazado" y enviar una notificación al empleado con los motivos del rechazo.
* **Registro de Actividades**: El flujo debe registrar todas las actividades en una lista de SharePoint o tabla de Dataverse para auditoría. Esto incluye la fecha y hora de la solicitud, la aprobación o rechazo, y los comentarios del gerente.
* **Condiciones y Manejo de Errores**: El flujo debe manejar condiciones específicas, como solicitudes incompletas o errores en la conexión con SharePoint. Debe incluir pasos de manejo de errores para notificar al administrador del sistema en caso de fallos.
* **Documentación**: proporcionar una documentación detallada del flujo, incluyendo capturas de pantalla de cada paso, explicaciones de las decisiones tomadas y cualquier configuración especial utilizada.

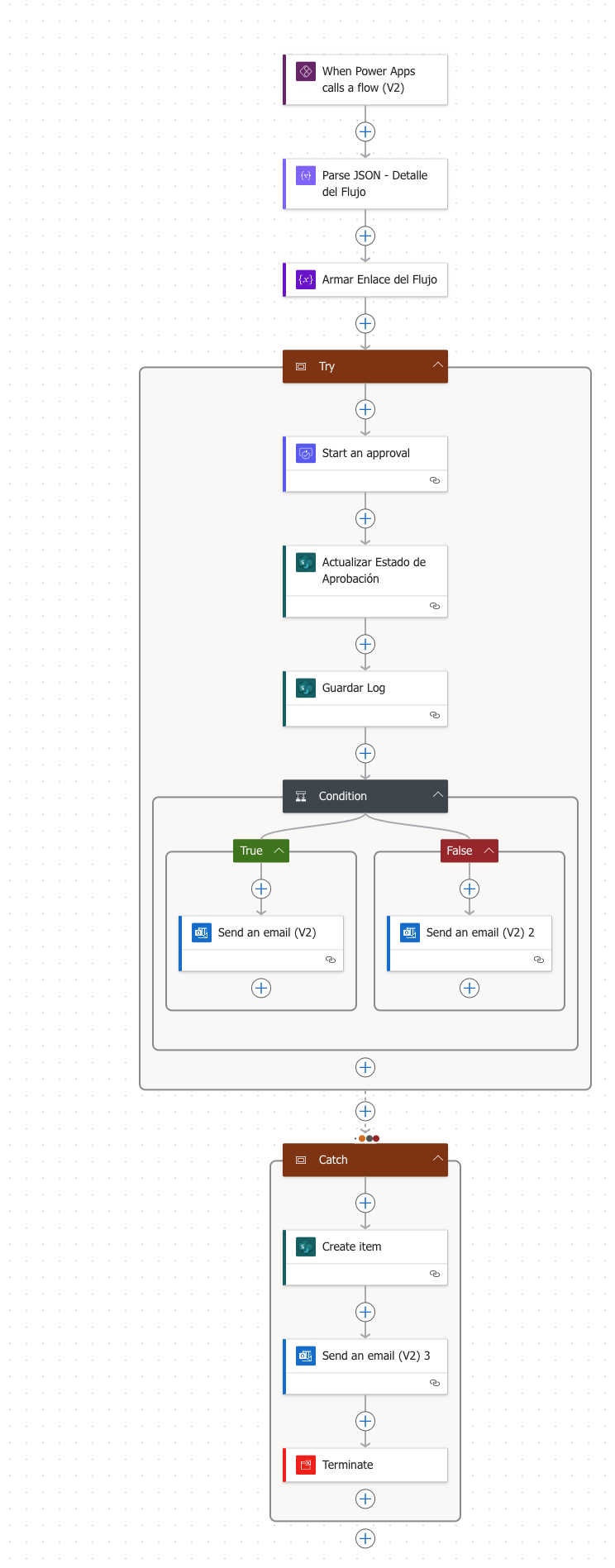
Entrega un video de no más de 20 minutos explicando los resultados y técnicas utilizadas. En el video responde lo siguiente:

En la ruta de GitHUb <https://github.com/mincho77/Fraude_Transacciones> hay un archivo llamado Aprobación de Gastos.mp4 que contiene la app

1. ¿Cómo manejaste la autenticación y autorización para asegurar que solo los gerentes puedan aprobar o rechazar solicitudes? Se creó una tabla para agregar a los gerentes, en ella están solo los correos de cada uno, con ello se garantiza que solo al gerente le llegue la solicitud que se creó por parte del empleado.
2. ¿Qué métodos utilizarías para optimizar el rendimiento del flujo y la app, especialmente si hay un gran volumen de solicitudes de gastos? De igual forma que se mencionó al principio Filtrar datos en el origen utilizando los conectores evitando traer datos innecesarios. Flujo que consulta txs masivas, filtro por fechas relevantes desde base de datos, para el caso DAtaverse. Seleccionar la opción de opción de paralelismo en los ciclos dentro del flujo para poder ejecutar varios hilos a la vez.
3. ¿Cómo implementarías pruebas unitarias para asegurar que el flujo y app funcione correctamente en diferentes escenarios? Modificando registros del archivo y verificando como lo mostré en el video la ejecución del flujo, dado que no se asocia internamente a la power app sino que se crea asociado a las tablas de dataverse o si fuese el caso a sharepoint. Verificando asi las dos tablas y los resultados después de recibir correos y aprobar cada uno.

Documentación Flujo Enviar Solicitud Aprobación:

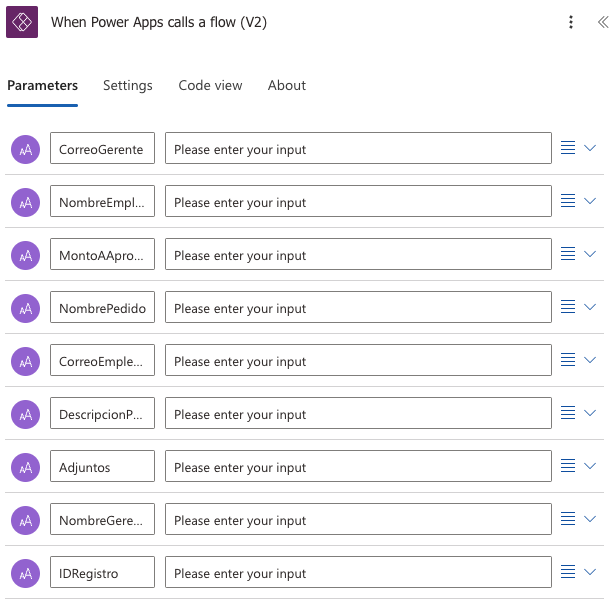
Este flujo tiene como objetivo procesar una solicitud proveniente de Power Apps, realizar una aprobación, actualizar el estado según el resultado de la aprobación y enviar correos electrónicos para notificar a los usuarios involucrados. También se asegura de registrar errores y terminar de forma controlada en caso de fallas.



Detalle del flujo completo:

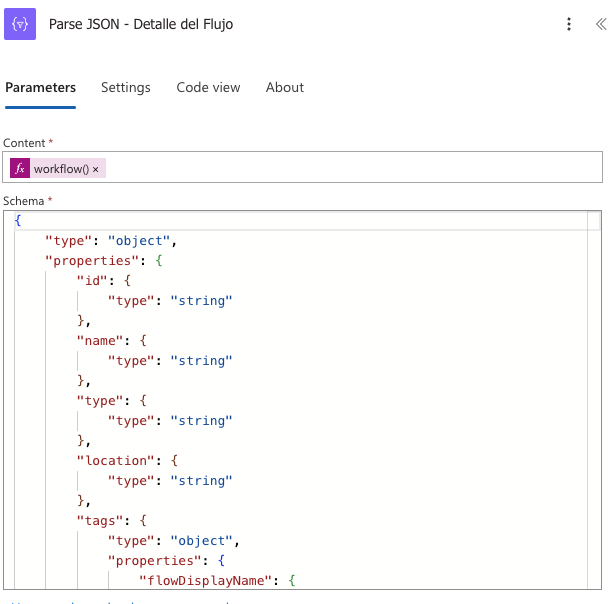
Variables enviadas desde la POwer App

Utilizo los siguientes elementos enviados desde la PowerApp para que sean utilizados dentro del Flujo, de esta forma tengo toda la información necesaria para el procesamiento de la información



**Parse JSON – Detalle del Flujo:**

Mediante esta actividad proceso el nombre del flujo y armo una trama que será utilizada posteriormente para poder enviar un enlace a quien controle los flujos y tenga facilidad mediante un link profundo de identificar en cual flujo se presentó el error en caso de que se de



En la siguiente variable me encargo de armar el link profundo del flujo que tengo agregando los siguiente 3 elementos:

body('Parse\_JSON\_Workflow\_details')?['tags']?['environmentName']

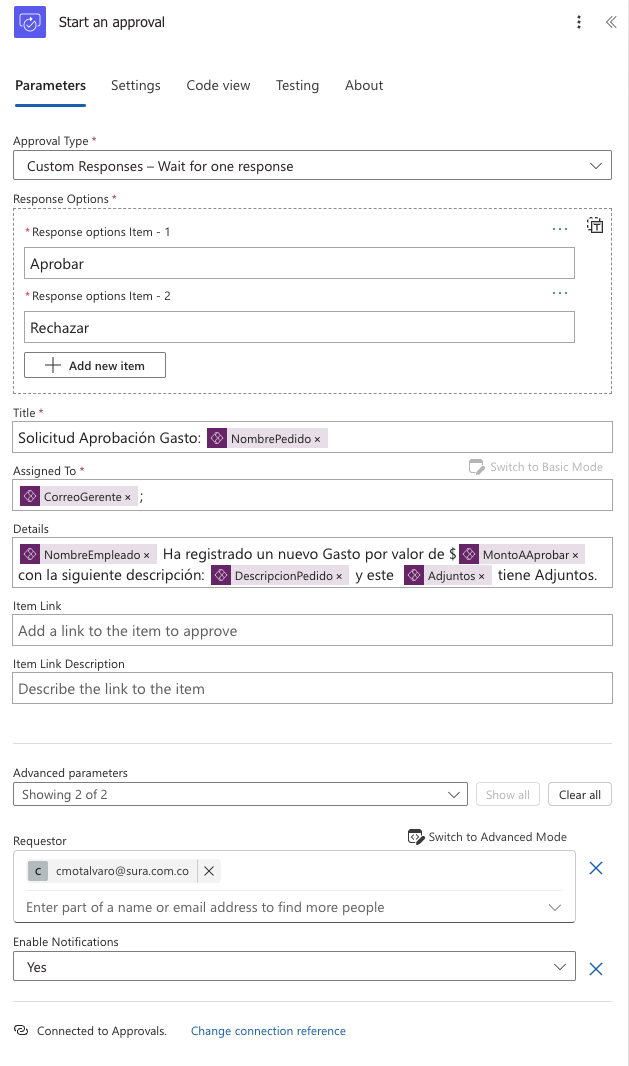
body('Parse\_JSON\_Workflow\_details')?['tags']?['logicAppName']

body('Parse\_JSON\_Workflow\_details')?['run']?['name']

 Agregándoles el nombre del JSON que contiene el nombre del flujo: Detalle\_del\_Flujo



**Detalle del Paso: Start an approval**

****

Este paso inicia un proceso de aprobación en el cual el gerente (definido por el parámetro CorreoGerente) debe decidir si aprueba o rechaza un gasto. Se proporcionan detalles del gasto en el mensaje de aprobación.

Tipo de Aprobación: Custom Responses - Wait for one response

Opciones de Respuesta: Aprobar, Rechazar

Título del Mensaje: Solicitud Aprobación Gasto: [NombrePedido]: Donde NombrePedido es un parámetro dinámico que representa el nombre del gasto a aprobar.

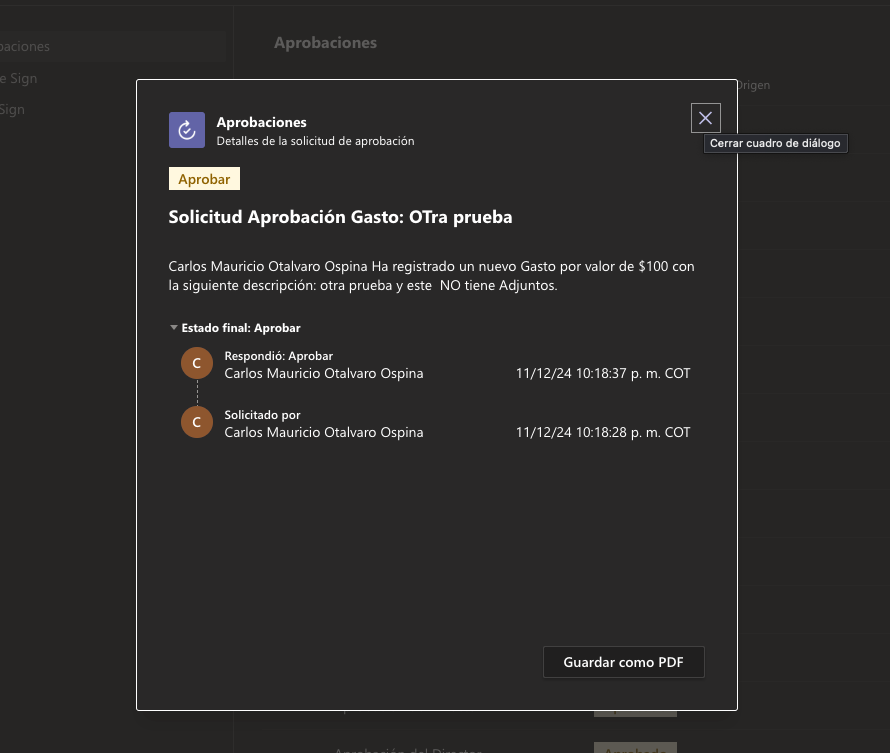
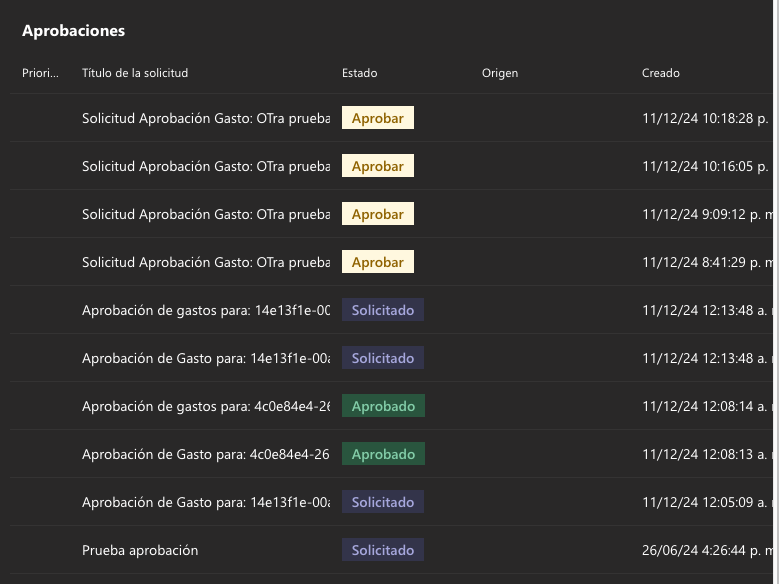
Asignado a: Dirección de correo electrónico del gerente (CorreoGerente) encargado de realizar la aprobación.

Mensaje: [NombreEmpleado] ha registrado un nuevo gasto por valor de $[MontoAprobar] con la siguiente descripción: [DescripcionPedido] y este [Adjuntos] tiene adjuntos.

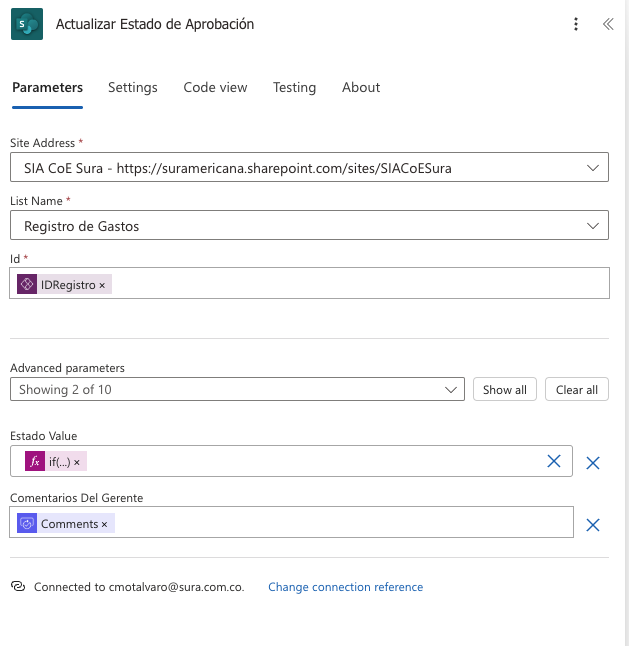
Cuando se activa este paso:

1. Se envía una solicitud de aprobación al correo electrónico definido en CorreoGerente.
2. El gerente recibe el título, los detalles del gasto, y puede tomar una decisión seleccionando una de las opciones de respuesta: **Aprobar** o **Rechazar**.
3. La respuesta es procesada en pasos posteriores del flujo.

Una vez el Gerente recibe la solicitud de aprobación esta es enviada al modulo de aprobaciones y allíes aprobada o rechazada, una vez se ejecute esta acción, el flujo continua en el paso de actualización de estado de aprobación .

**Actualizar Estado de Aprobación**



Este paso actualiza un registro en una lista de SharePoint llamada **"Registro de Gastos"**. Es utilizado para reflejar el estado de una solicitud de aprobación y registrar comentarios proporcionados por el gerente.

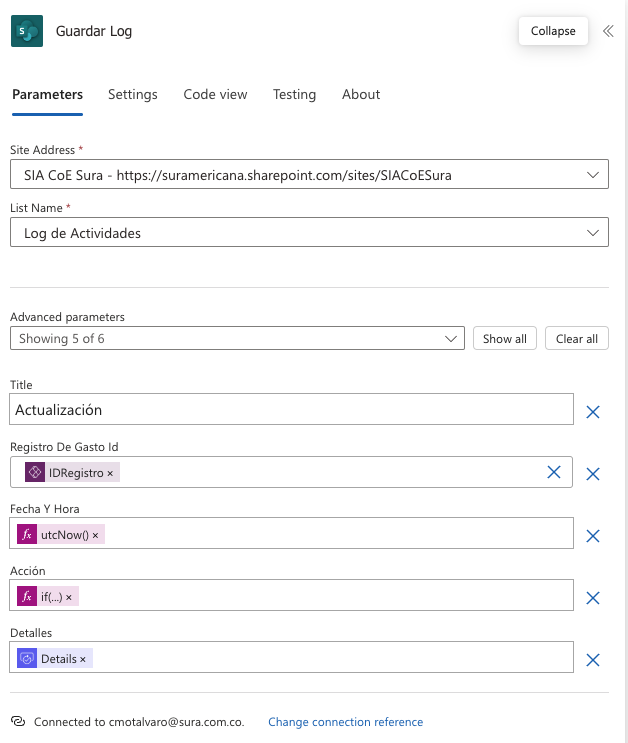
Este paso asegura que los resultados de la aprobación se reflejen directamente en la lista de SharePoint asociada con el sistema de gestión de gastos.

El campo de estado del registro (Estado Value) se ajusta en función de la decisión del gerente.

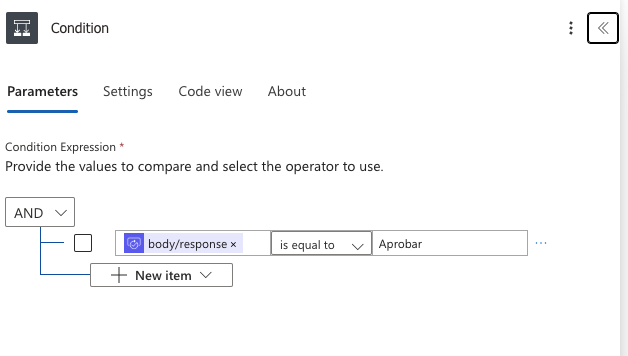
Los comentarios proporcionados por el gerente durante el proceso de aprobación se almacenan en el campo Comentarios Del Gerente.

**Guardar Log**

Dado que se requiere tener guardada información tanto en la creación como en la actualización de estados, una vez se aprueba el estado y se modifica el mismo en la tabla de solicitudes, también se almacena información en la tabla de Log de Actividades, con ello se tiene información completa de lo que se realizó, para cada uno de los registros en el log de actividades siempre se tiene un campo que identifica qué se hizo, si fue una creación de un registro, una Actualización o si fue un Error generado durante el proceso.

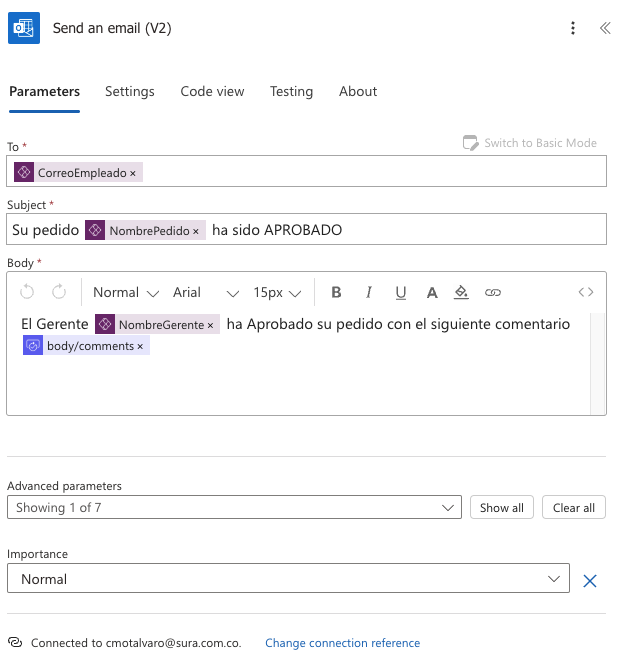


**Condición después de la aprobación**

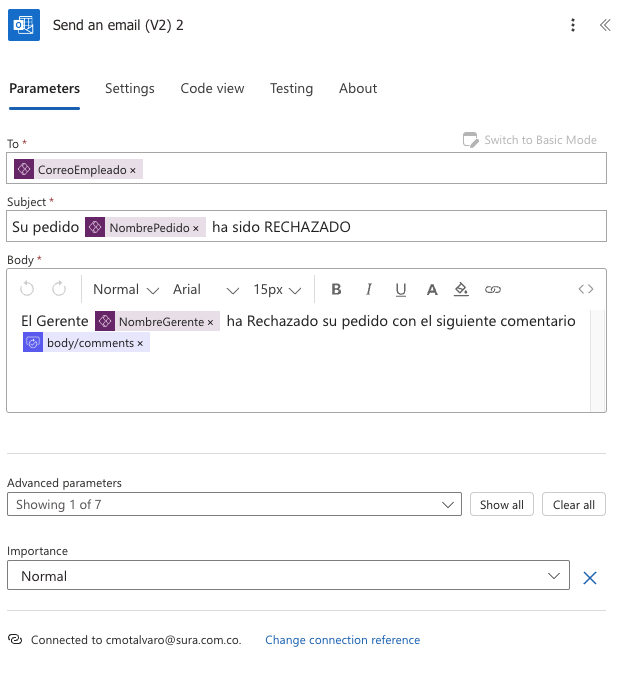
****

Una vez el gerente Aprueba o rechaza la solicitud se identifica la respuesta y se envía la notificación al empleado.

**Mensaje Aprobación**

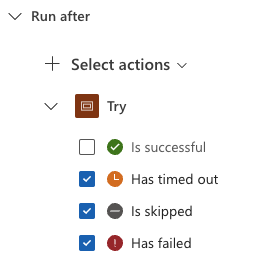
****

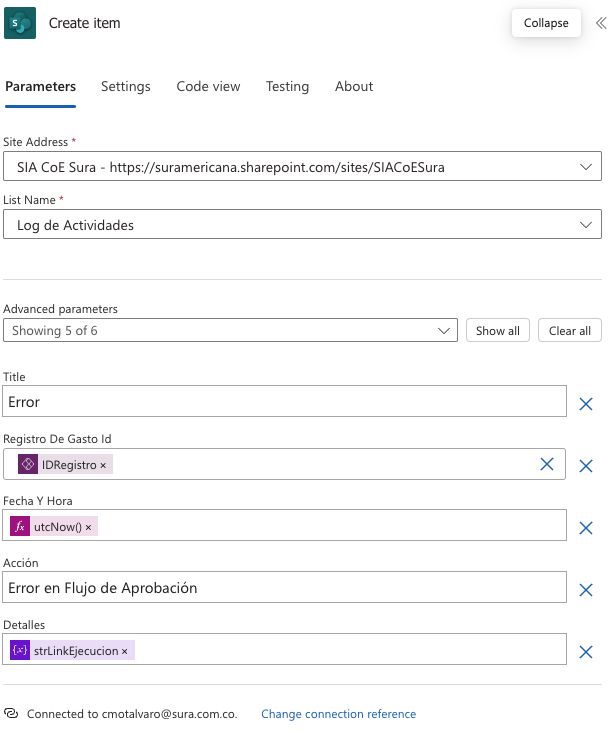
**Mensaje Rechazo**

****

**Control de Errores**

Se utilizaron dos scope uno para Try y otro para Catch. En el primero se intenta ejecutar el flujo normalmente, en caso de fallar después de la ejecución:





Se procede con la creación de un item adicional en la lista de sharepoint de Log, se envía un correo con el link profundo al administrador del flujo y se Termina el flujo con un estado fallido para que no se vea dentro de la lista del mismo como si se hubiese ejecutado exitosamente por haber terminado.

