Centro Universitario de Ciencias Exactas e Ingeniería

Computación Tolerante a Fallas

Workflow Managers

MICHEL EMANUEL LOPEZ FRANCO

Juan Pablo Hernández Orozco

219294285

Objetivo:

¿Que es Prefect?

Introducción a PREFECT
Task Failed Successfully

- Jeremiah Lowin

https://youtu.be/TlawR_gi8-Y?list=PLMGWGsnelbxcHmA5cVRq8a39S9s_gxAMq

Tarea:

- Parte 1.
  Seguir el tutorial siguiente y ejecutarlo en tu maquina.

Getting Started with Prefect (PyData Denver)

https://www.youtube.com/watch?v=FETN0iivZps&t=2545s

- Parte 2.
  Modificarlo y Generar un ejemplo

link nuevo
https://jsonplaceholder.cypress.io/

Desarrollo:

¿Qué es Prefect?

Prefect es una plataforma moderna para la orquestación de flujos de trabajo que te permite definir, ejecutar y monitorear pipelines de datos en Python de manera sencilla y robusta. En esencia, Prefect se enfoca en dos aspectos clave: Ingeniería positiva vs. ingeniería negativa:

Te permite escribir el código principal de tu lógica (ingeniería positiva) mientras se encarga del manejo de errores y fallos (ingeniería negativa). Esto significa que, en lugar de detener todo el proceso ante un error, Prefect “falla exitosamente”, registrando y gestionando los fallos de manera controlada para que puedas actuar sobre ellos sin interrumpir todo el flujo.

Definición y ejecución de tareas: A través de decoradores (@task) y la construcción de flujos (Flows), Prefect te permite crear pipelines donde cada tarea tiene un estado monitorizado. Esto facilita tanto la ejecución local como la distribuida, mejorando la visibilidad del desempeño de las tareas dentro del flujo.

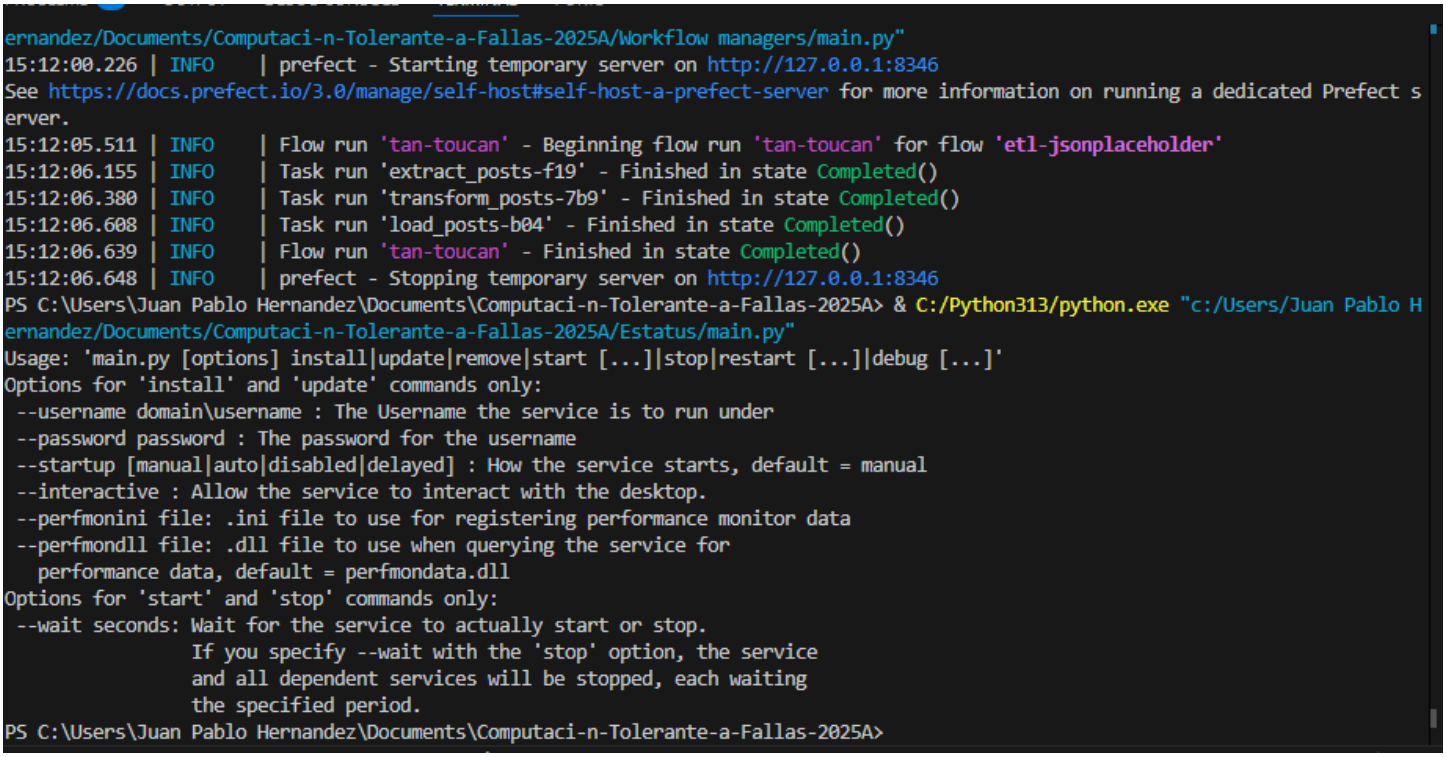
En otras palabras, Prefect fue diseñado para superar algunas limitaciones de herramientas anteriores (como Apache Airflow), ofreciendo un sistema en el que cada tarea es un “bloque negro” que se ejecuta, se le asigna un estado (éxito, fallo, etc.) y ese estado se utiliza para decidir los siguientes pasos. Esta capacidad de gestionar el estado y los errores de manera granular permite diseñar flujos de trabajo más resilientes y adaptados a las necesidades reales de los procesos de datos. Por ejemplo, en el tutorial “Getting Started with Prefect” se ilustra cómo construir un pipeline ETL básico, donde se definen tareas para extraer datos, transformarlos y cargarlos en una base de datos, utilizando el enfoque de Prefect para gestionar cada paso y sus posibles errores.

Parte 1:

Ahora veremos el resultado del tutorial corriendo en mi pc:

```
PS C:\Users\Juan Pablo Hernandez\Documents\Computaci-n-Tolerante-a-Fallas-2025A> & C:/Python313/python.exe "c:/Users/Juan Pablo Hernandez/Documents/Computaci-n-Tolerante-a-Fallas-2025A/Workflow managers/main.py"
14:52:27.800 | INFO    | prefect - Starting temporary server on http://127.0.0.1:8303
See https://docs.prefect.io/3.0/manage/self-host#self-host-a-prefect-server for more information on running a dedicated Prefect server.
14:52:32.473 | INFO    | Flow run 'fluffy-dodo' - Beginning flow run 'fluffy-dodo' for flow 'etl-jsonplaceholder'
14:52:33.126 | INFO    | Task run 'extract_posts-0a2' - Finished in state Completed()
14:52:33.344 | INFO    | Task run 'transform_posts-aab' - Finished in state Completed()
14:52:33.572 | INFO    | Task run 'load_posts-4f6' - Finished in state Completed()
14:52:33.591 | INFO    | Flow run 'fluffy-dodo' - Finished in state Completed()
14:52:33.598 | INFO    | prefect - Stopping temporary server on http://127.0.0.1:8303
PS C:\Users\Juan Pablo Hernandez\Documents\Computaci-n-Tolerante-a-Fallas-2025A> & C:/Python313/python.exe "c:/Users/Juan Pablo Hernandez/Documents/Computaci-n-Tolerante-a-Fallas-2025A/Workflow managers/tutorial.py"
14:52:47.989 | INFO    | prefect - Starting temporary server on http://127.0.0.1:8553
See https://docs.prefect.io/3.0/manage/self-host#self-host-a-prefect-server for more information on running a dedicated Prefect server.
14:52:52.669 | INFO    | Flow run 'olive-koala' - Beginning flow run 'olive-koala' for flow 'etl-flow'
14:52:53.009 | INFO    | Task run 'get_complaint_data-9d4' - Finished in state Completed()
14:52:53.244 | INFO    | Task run 'parse_complaint_data-d6f' - Finished in state Completed()
14:52:53.460 | INFO    | Task run 'store_complaints-0c7' - Finished in state Completed()
14:52:53.488 | INFO    | Flow run 'olive-koala' - Finished in state Completed()
14:52:53.496 | INFO    | prefect - Stopping temporary server on http://127.0.0.1:8553
PS C:\Users\Juan Pablo Hernandez\Documents\Computaci-n-Tolerante-a-Fallas-2025A>
* History restored
```

Parte 2:



Conclusión:

Prefect se presenta como una solución innovadora para la orquestación de flujos de trabajo en Python. Su diseño permite separar la lógica principal del manejo de errores, lo que se traduce en un enfoque de "ingeniería positiva vs. ingeniería negativa": mientras el desarrollador se concentra en la lógica esencial, la plataforma se encarga de gestionar y registrar los fallos sin interrumpir todo el proceso. Además, al utilizar decoradores y construir flujos con tareas monitorizadas, Prefect facilita tanto la ejecución local como la distribuida, ofreciendo una visibilidad detallada y un control granular del estado de cada tarea. Esta capacidad de "fallar exitosamente" y adaptarse a diversas necesidades lo posiciona como una alternativa robusta y flexible frente a herramientas anteriores, como Apache Airflow, siendo especialmente útil en la implementación de pipelines ETL y otros procesos de datos complejos.

Repositorio:

https://github.com/ELJuanP/Computaci-n-Tolerante-a-Fallas-2025A/tree/main/Workflow%20managers