



FACULTAD DE CIENCIAS

UNAM

# MODELADO Y PROGRAMACIÓN

## PROYECTO 1: "WEB SERVICE"

Cruz Campos José Eduardo & Martínez Herrera Miguel Agustín

# 1. Definición del problema

Para este proyecto se nos indica realizar un programa que funcione para el aeropuerto de la CDMX, este requiere informes acerca del clima respecto a las ciudades de salida o llegada, esto para hacerlo funcionar para tres mil tickets, los cuales se dan el mismo día que el algoritmo se necesita. Para poder realizarlo se nos dio un documento csv que contiene información necesaria para realizar el proyecto, con esto podemos saber que lo único que va a realizar el usuario es ingresar las iniciales de la ciudad que quiere consultar.

# 2. Análisis del problema

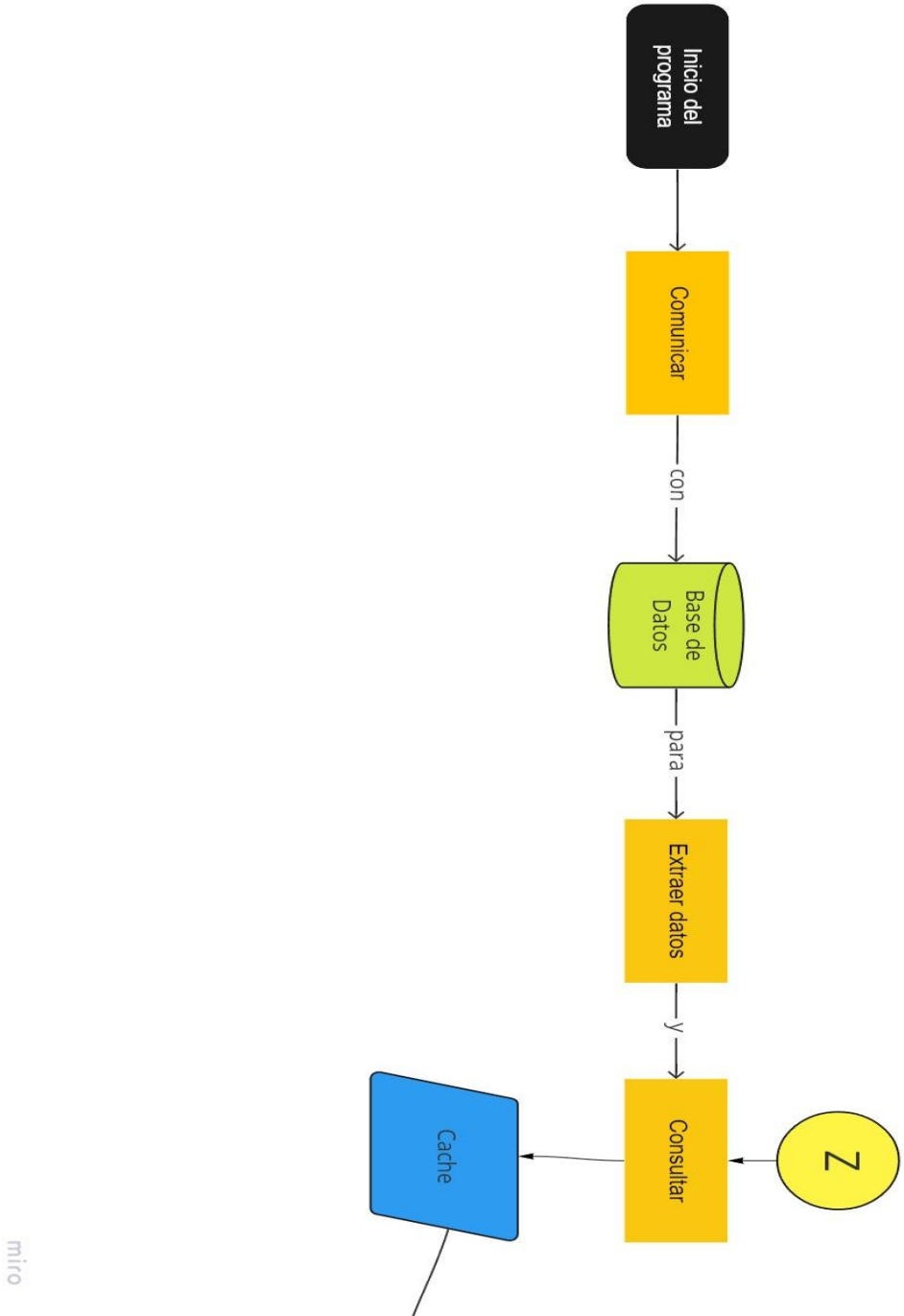
Para lograr consultar el clima de cada urbe es necesario conocer la manera en que se debería realizar las denominadas al API que se va a escoger, por lo cual podría utilizarse el nombre de cada urbe para estas denominadas, no obstante, en el documento que nos brindan, este nombre viene abreviado. De esta forma vamos a tener que la mejor forma será usar las coordenadas de cada localidad para hacer las consultas, esto podría ser más que suficiente. Además, al hacer el algoritmo de este plan es importante tener en cuenta que el web service que usemos se limita a proveer cierta proporción de información por minuto, por lo cual es importante que las metrópolis que permanecen reiteradas en la base de datos no vuelvan a hacer una demanda. Además, se debe tener presente que el clima de cada localidad debería comprender los rubros de temperatura, temperatura mínima, temperatura máxima, humedad y sensación térmica. En resumen, es necesario leer el documento csv, hacer las demandas del clima al web service, procesar la información devuelta y regresar los datos al usuario de la forma más amistoso y eficiente viable.

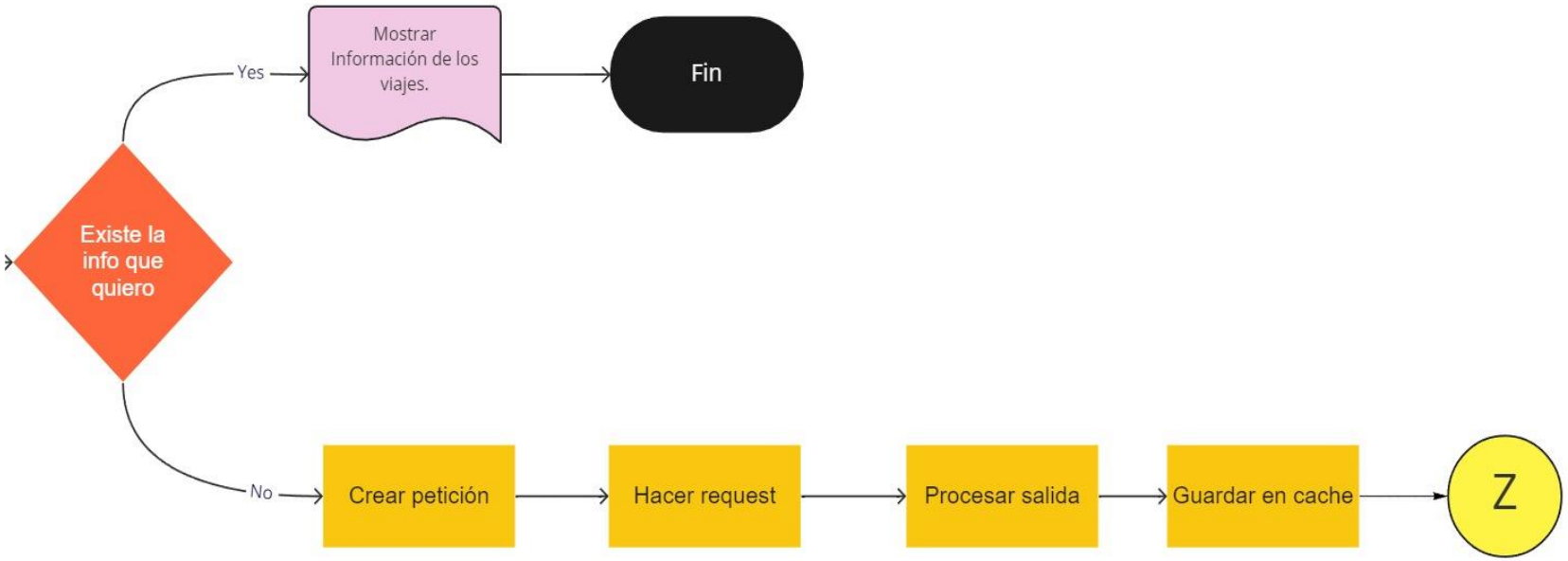
# 3. Selección de la mejor alternativa

El proyecto será realizado en Python puesto que es un lenguaje que nos es mas amigable y nos permite de alguna manera manejar la información de una manera más útil, además de que podemos usar los diccionarios. Tenemos que también funciona en cualquier sistema operativo debido a que es multiplataforma. Para hacer las consultas del clima usaremos el web service denominado: OpenWeather, el cual nos provee de una API que es eficiente y nos otorga la información que requerimos. La mejor manera de resolver el problema de que las localidades se repetirán en la base de datos, ocasionando de esta forma que se hicieran denominadas al API innecesarias y el programa tardará más tiempo en este proceso. Escogió llevar a cabo un caché, el cual debido a una lista en la que se almacenan objetos con la información del clima de cada urbe, se proviene a comprobar anterior a que se haga una llamada, ocasionando que si dicha urbe ya había pasado antes, se regrese la

información del sitio y no se hagan más demandas, hasta que aparezca una totalmente nueva urbe. Además, escogió que la mejor forma de enseñar el clima de la urbe de procedencia y de llegada podría ser por medio de una interfaz gráfica, que posibilita una adecuada interpretación de los datos del cliente que usé el programa.

## 4. Diagrama de flujo




Yes
Mostrar Información de los viajes.
Fin
Existe la info que quiero
No
Crear petición
Hacer request
Procesar salida
Guardar en cache
Z

Para poder hacer funcional este algoritmo, primero vamos a crear una base de datos a partir de un diccionario el cual está basad en diversos atributos como el nombre de la ciudad y sus coordenadas. Posteriormente, se hace un recorrido por este diccionario para así poder ir consultando el clima de cada ciudad a través de llamadas al API. También tenemos que para cada petición, esa ciudad consultada se irá almacenando, así que cada llamada al API se va a dirigir a un filtro al cual nombraremos el caché, que nos dice si esa ciudad ya había sido consultada. Esto, gracias a que se comparamos el primer diccionario que hicimos con el contenido de las ciudades que se irán guardando y con esto vamos a poder saber si la ciudad próxima a consultar ya había pasado con anterioridad y así devolver los resultados correspondientes. Posteriormente, cuando se tiene toda la información de los viajes, y procederemos a dar la información al usuario que esté haciendo las consultas.

## 5. Mantenimiento que va a requerir en un futuro y precio del proyecto.

El mantenimiento que va a necesitar este programa es el ir actualizando cómo es que este lee y procesa la información de la base de datos, debido que puede existir la probabilidad de que en un futuro los datos del csv se encuentren diferentes, provocando que se ocupe una actualización rápida del programa. Otro aspecto fundamental es el de expandir la proporción de denominadas al API, si se necesita consultar una porción más grande del clima de las localidades en una época definido, por lo cual podría ser primordial llevar a cabo una estrategia distinta o adquirir un paquete diferente en la API que estamos consultando la cual OpenWeather. Además, el mantenimiento puede integrar novedosas funcionalidades como el de que la interfaz sea accesible y amistoso a personas que hablan un lenguaje distinto, que se añada novedosas herramientas para facilitar la averiguación de el clima en la urbe, entre otras cosas. Por este algoritmo ya aplicado cobraríamos entre $150 y $250 dólares basándonos en las necesidades del cliente. Y dependiendo del mantenimiento podría ir desde los $15 a los $35 dólares .