REST API Y FLASK

Características principales:
Ligero y minimalista.
Compatible con JSON para manejar datos.
Soporte para rutas y endpoints personalizados.
Extensiones disponibles para funcionalidades avanzadas (e.g., Flask-RESTful).

FLASK RESY APYTHON

Definición:
Un microframework de Python para crear aplicaciones web y REST APIs de forma sencilla y flexible.

Flask
web development, one drop at a time

COMPONENTES CLAVE:

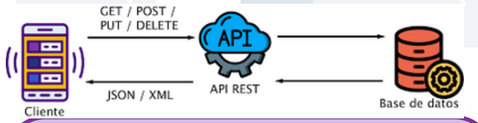
- Endpoints: Definidos con decoradores (@app.route).
- HTTP Methods: Se configuran explícitamente (GET, POST, PUT, DELETE).
- JSON Responses: Uso de flask.jsonify() para devolver datos estructurados.

Pasos básicos para crear una REST API:

1. Instalar Flask:

```
bash
pip install flask
```

Ventajas:
Fácil de aprender y configurar.
Flexible para personalizar.
Ideal para aplicaciones pequeñas y medianas.



Componentes clave:
- Recursos: Representados como URLs únicas.
- HTTP Methods: Uso de verbos estándar (GET, POST, PUT, DELETE).
- Representaciones: Respuestas en formatos como JSON, XML, o HTML.

ARQUITECTURA REST API

- Un estilo arquitectónico para construir servicios web basados en recursos.
- Características principales:
- Cliente-Servidor: Separación entre cliente (front-end) y servidor (back-end).
- Stateless: Cada solicitud es independiente; el servidor no almacena información de sesión.

COMO FUNCIONA?

Cliente
Envia una soolicitud al servidor
Recibe la solicutd y regresa una respuesta

Sin estado
El servidor es indepentiente ya que no recuerda la petición y vuelve a solicitar la info.

CÓDIGOS DE RESPUESTA

Son códigos HTTP que indican el resultado de una solicitud al servidor.

Clasificación:
- 1xx - Informativos:
Ejemplo: 100 Continue (Solicitud recibida, espera respuesta).
- 2xx - Éxito:
Ejemplo: 200 OK (Solicitud exitosa).
201 Created (Recurso creado).
- 3xx - Redirección:
Ejemplo: 301 Moved Permanently (URL redirigida permanentemente).
302 Found (Redirección temporal).
-

- 4xx - Errores del cliente:
Ejemplo: 400 Bad Request (Solicitud mal formada).
401 Unauthorized (Falta autenticación).
404 Not Found (Recurso no encontrado).
- 5xx - Errores del servidor:
Ejemplo: 500 Internal Server Error (Error en el servidor).
503 Service Unavailable (Servicio no disponible).

PRINCIPALES MÉTODOS HTTP

GET
Propósito: Obtener datos de un recurso.
Ejemplo:
Solicitud: GET /users
Respuesta: [{"id": 1, "name": "John"}]
POST
Propósito: Crear un nuevo recurso.

PUT
- Propósito: Actualizar un recurso existente (reemplazo completo).
PATCH
- Propósito: Modificar parcialmente un recurso.

DELETEPropósito: Eliminar un recurso.Ejemplo:
Solicitud: DELETE /users/1
Respuesta: 204 No Content