Types of parameters:

1. Query Parameters: Sent in query as example :http://example.com/resource?param1=value1¶m2=value2
   Request methods : POST, PUT, PATCH
   format: JSON, form data, etc.
   To access the query parameters we use :a) In django : request.get
   b)In Flask : request.args
   c) In fast : defined in function parameters with 'Query'

2. Body Parameters: Body parameters are typically used for data that is sent in the body of an HTTP request. This data is often associated with the request payload and is commonly used in operations like creating or updating a resource.
   Request methods : GET
   format: Appended to the URL as key-value pairs
   To access the body parameters we use :a)In django :request.data
   b) In flask : request.json
   c) In fastapi:Defined using Pydantic models

	Body Parameters (Request Body)	Query Parameters (URL)
Django	`request.data`	`request.GET`
FastAPI	Defined using Pydantic models	Defined in function parameters with `Query`
Flask	`request.json`	`request.args`
Request Methods	POST, PUT, PATCH	GET
Format	JSON, form data, etc.	Appended to the URL as key-value pairs
Example	Creating or updating a resource	Filtering resources

Path Parameters:

- Path parameters are used to extract values from the URL path itself, allowing for dynamic and variable-based routing.

- In Django, path parameters are defined in the URL patterns using angle brackets `<parameter_name>`. They can be accessed as named parameters in the corresponding view function.
- In Flask, path parameters are defined in the route using angle brackets `<parameter_name>`. They can be accessed as function arguments or using the `request.args` dictionary-like object.
- In FastAPI, path parameters are defined within the route declaration using curly braces `{parameter_name}`. They can be directly used as function parameters in the corresponding path operation function.
- Path parameters are commonly used to identify specific resources or provide unique identifiers in the URL.