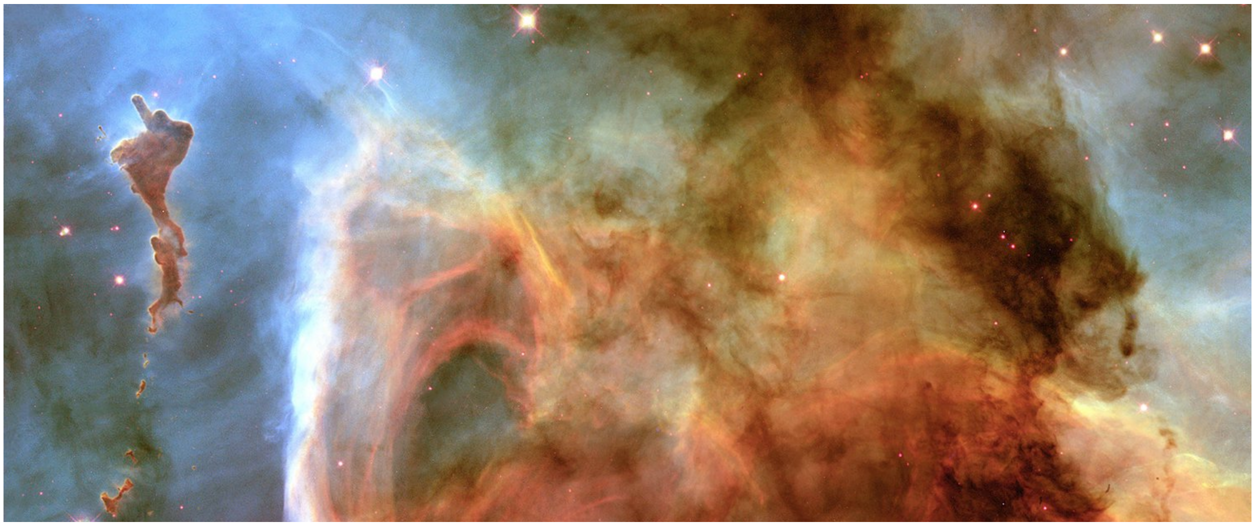

Request Body

Till now we have sent a URL request to the API and received response.

We get the response body.

To view this response body, you can visit the docs and make a request from there.

However, just like a response body we also have request body as well.

Request body is data sent by client to the API.

API always has to send a response body, but clients do not need to send request body all the time.

Just as we use a GET method to get data, to send data to an api, i.e to send request body we need POST method.

This post method is used when we have to submit the form data to the database.

Let's first make a simple post request.

```
@app.post('/adduser')
def adduser():
    return {'user':{'name':'Ashutosh','email':'ashutosh99@gmail.com'}}
```

Now to declare a request body, we can use pydantic data models.

Lets say we want to create an api which accepts user data via a form.

To do this, we need to save multiple user fields like name,email and age.

To accept this via an API POST request, we make use of a request body as we want to request data from the client.

To create this request body, we create a model.

To create this model we use Pydantic.

Pydantic is a Python library which allows us to define custom data type.

For example, right now we use str, float, int which are built in data types.

But to define our own types we use Pydantic.

Now to accept the user data, we create a Pydantic type of our own.

To create this, we need to create a user model.

Lets call that model as profile.

First we need to import base model from Pydantic.

```
from pydantic import BaseModel

class Profile(BaseModel):
    name: str
    email:str
    age:int
```

Now make use of this in a route to handle post request.

```
#create a post route
@app.post('/profile/')
def createprofile(profile:Profile):
    return profile
```

Now visit the docs page and try out the post route.

Try editing the post body and submit the request.

Analyse the response by the API.