# Web API-ASP.NET Core

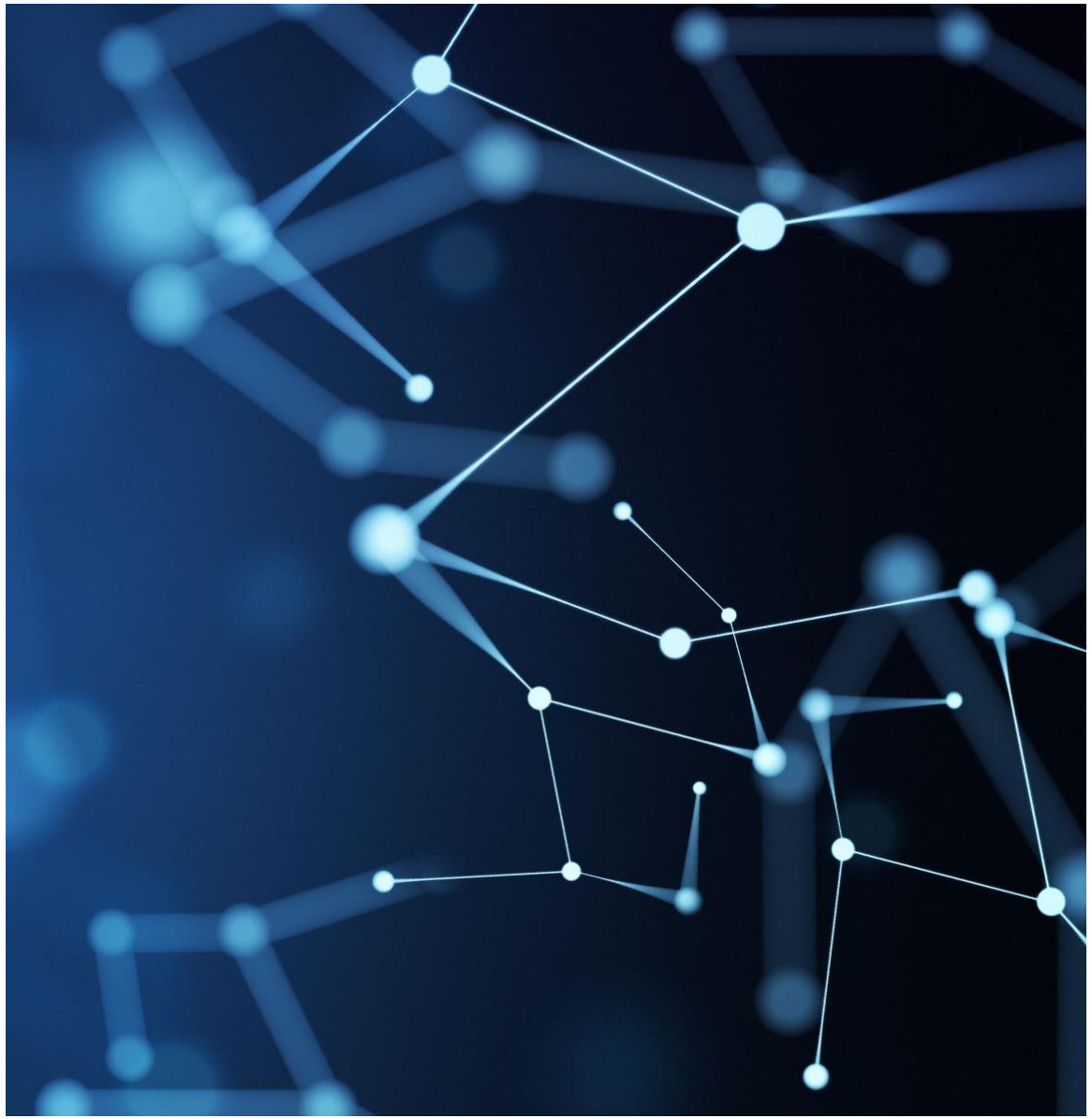
**CREATED BY** :- Deep Parmar

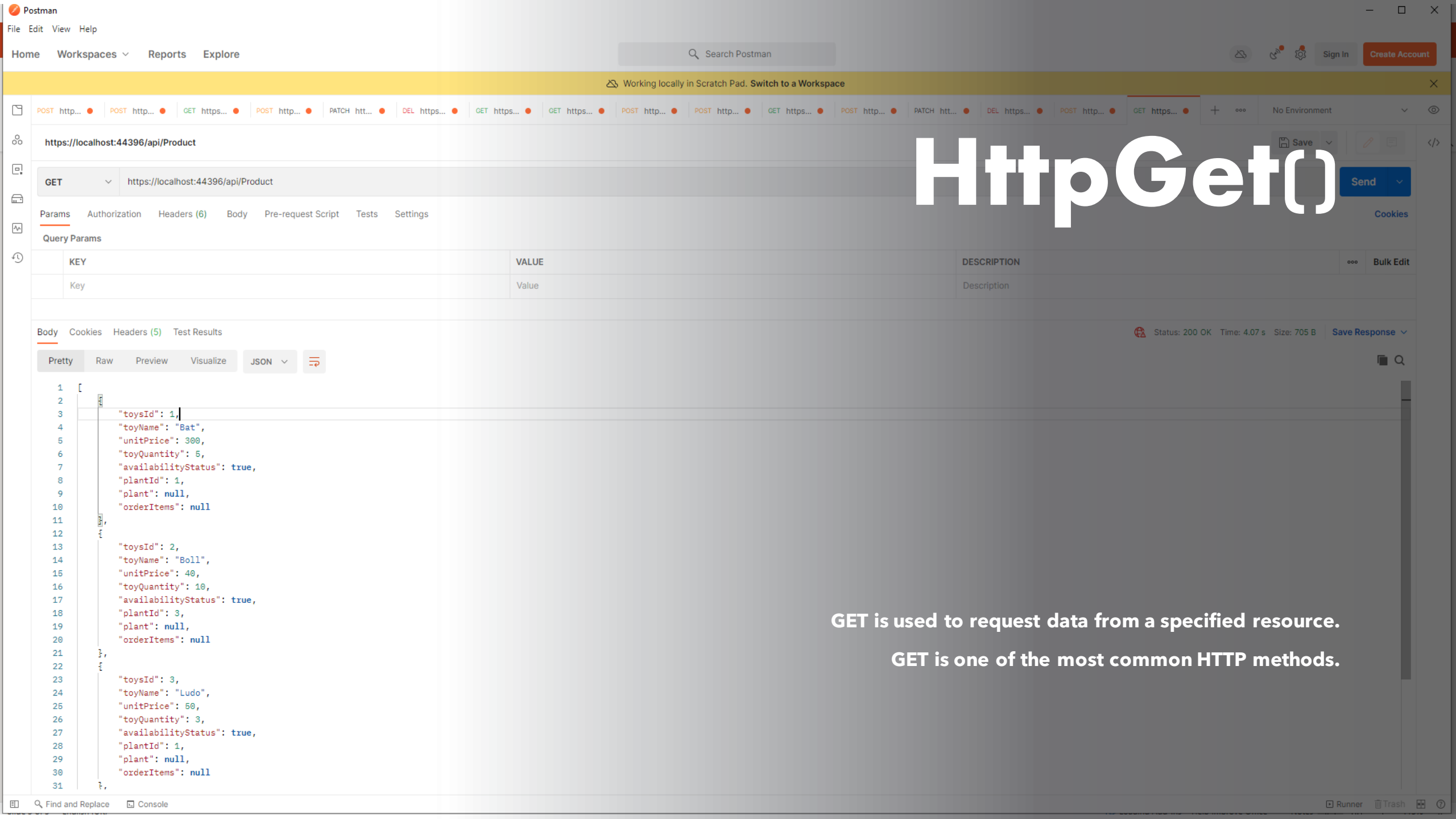
## Index

-HTTP Methods

-Passing Header

-Repository Pattern

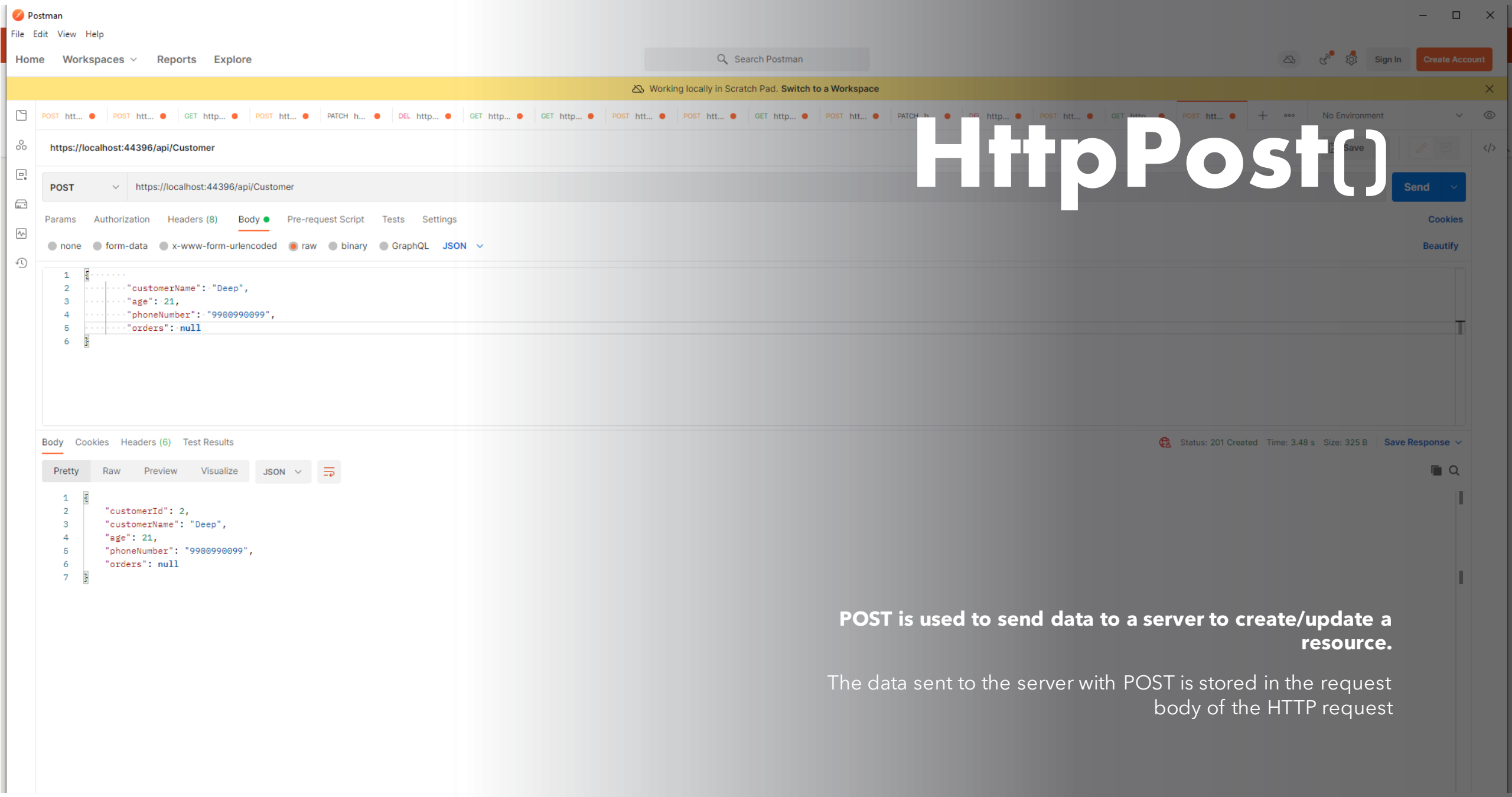





# HttpGet()

**GET is used to request data from a specified resource.**

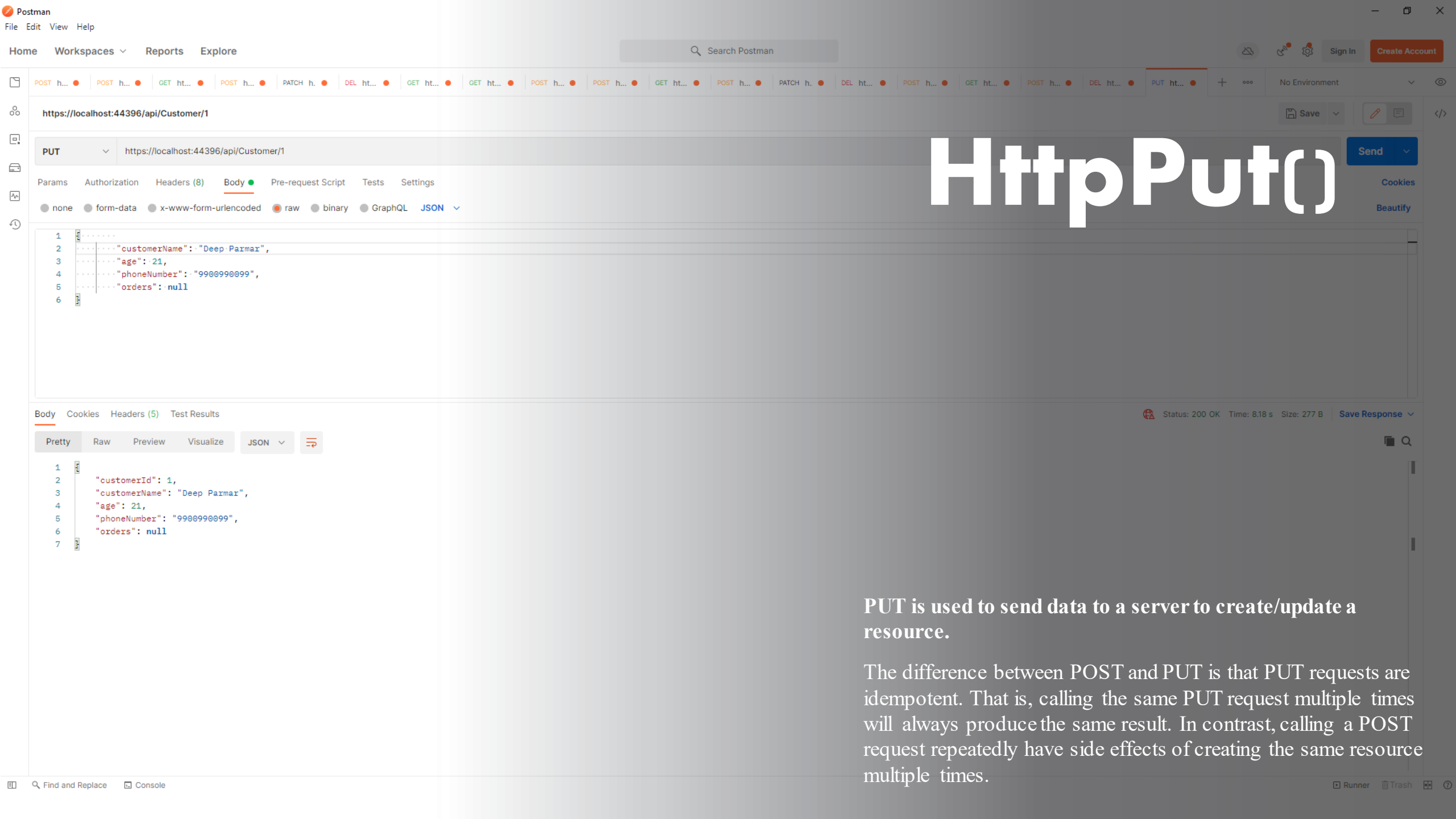
**GET is one of the most common HTTP methods.**




# HttpPost()

**POST is used to send data to a server to create/update a resource.**

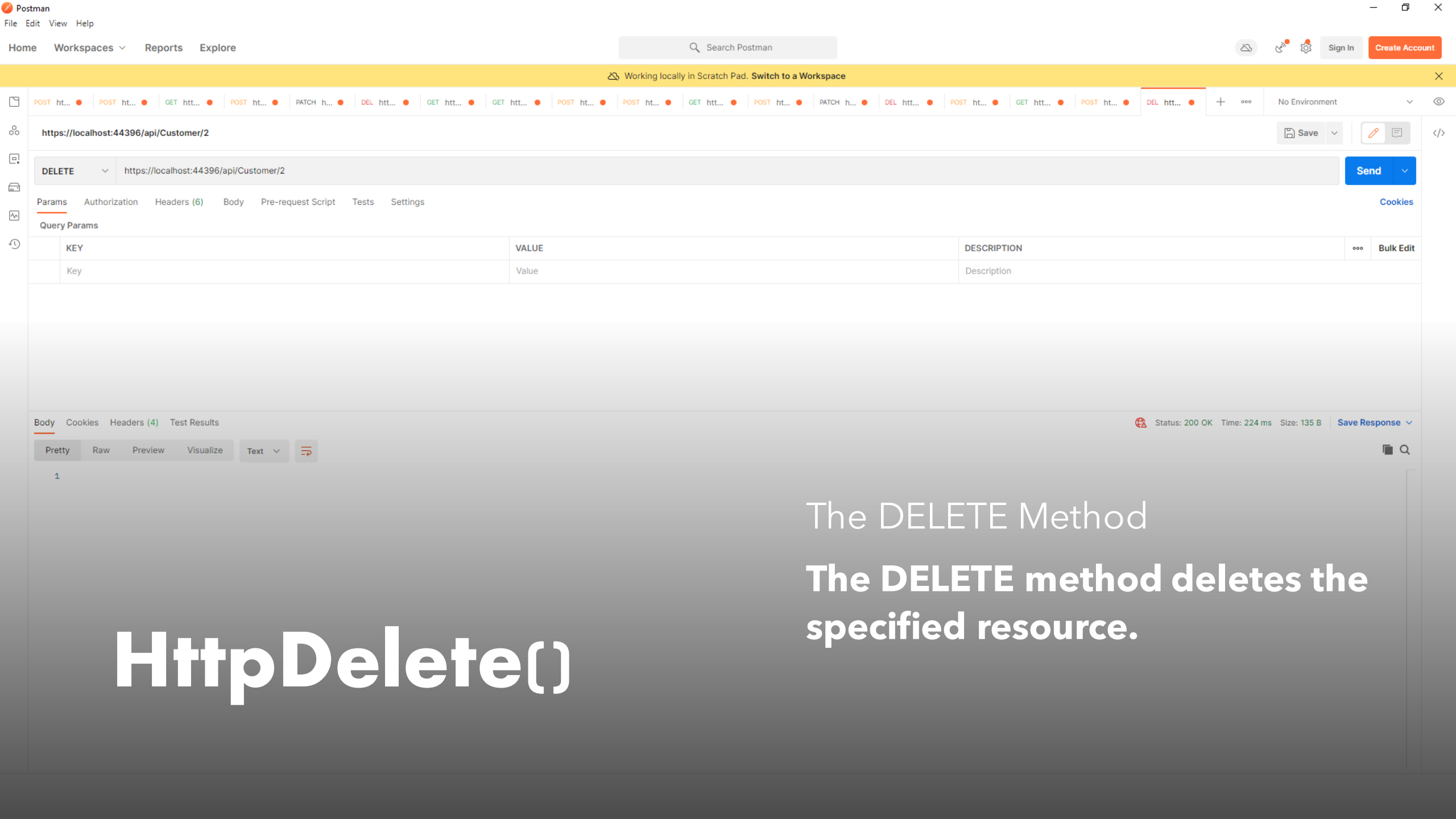
The data sent to the server with POST is stored in the request body of the HTTP request




# HttpPut()

**PUT is used to send data to a server to create/update a resource.**

The difference between POST and PUT is that PUT requests are idempotent. That is, calling the same PUT request multiple times will always produce the same result. In contrast, calling a POST request repeatedly have side effects of creating the same resource multiple times.


Postman
File Edit View Help
Home Workspaces Reports Explore
Search Postman
Sign In
Create Account
Working locally in Scratch Pad. Switch to a Workspace
No Environment
https://localhost:44396/api/Customer/2
Save
DELETE
https://localhost:44396/api/Customer/2
Send
Params Authorization Headers (6) Body Pre-request Script Tests Settings
Cookies
Query Params
KEY
VALUE
DESCRIPTION
Bulk Edit
Key
Value
Description
Body Cookies Headers (4) Test Results
Status: 200 OK Time: 224 ms Size: 135 B
Save Response
Pretty Raw Preview Visualize Text
1
HttpDelete()
The DELETE Method
The DELETE method deletes the specified resource.

# Passing Headers

API headers are **like an extra source of information for each API call you make**. Their job is to represent the meta-data associated with an API request and response.

# Repository Pattern

With the Repository pattern, we create an abstraction layer between the data access and the business logic layer of an application. By using it, we are promoting a more loosely coupled approach to access our data from the database. Also, the code is cleaner and easier to maintain and reuse. Data access logic is in a separate class, or sets of classes called a repository, with the responsibility of persisting the application's business model.