

API Chaining

API Chaining dediğimiz olay aslında bir response'un başka bir requestin inputu veya bir parçası olması. Örnek olarak mesela POST ile oluşturduğumuz bir veriyi GET ile çekip istenilen veri ile aynı mı diye kontrol etme aşaması POST'un response'u GET'in requestinde kullanılacak olması bir chaining işlemidir.

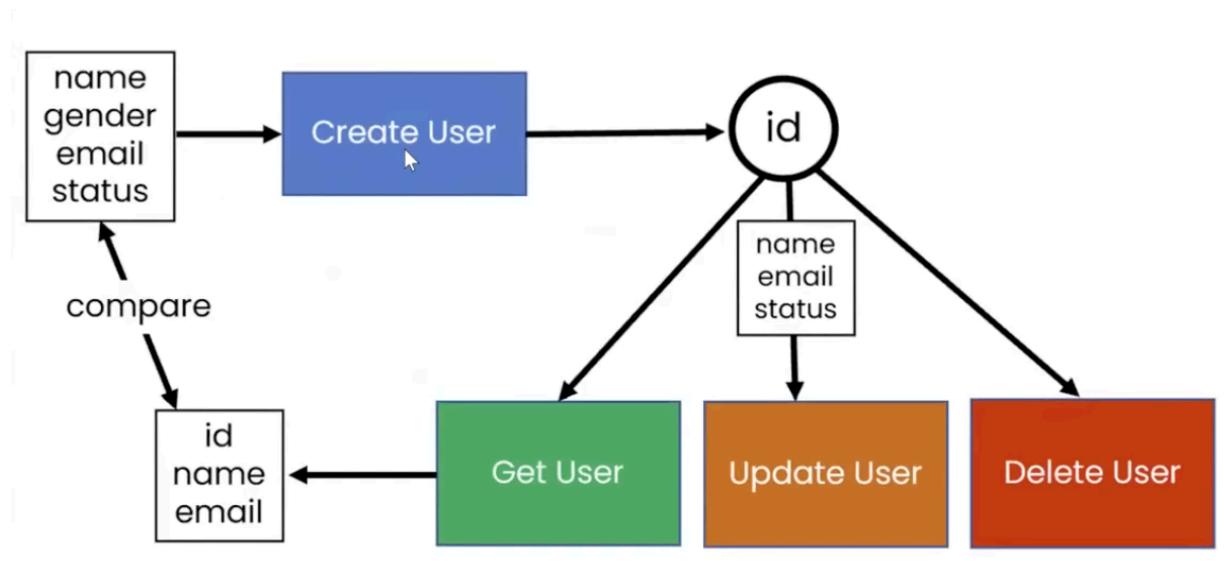
Remote API: Dışardan server'a eriştiğimiz API'lar

Local API: localhost kendi bilgisayarlığımızda bulunan ve sadece kendimizin erişebildiği API'ler

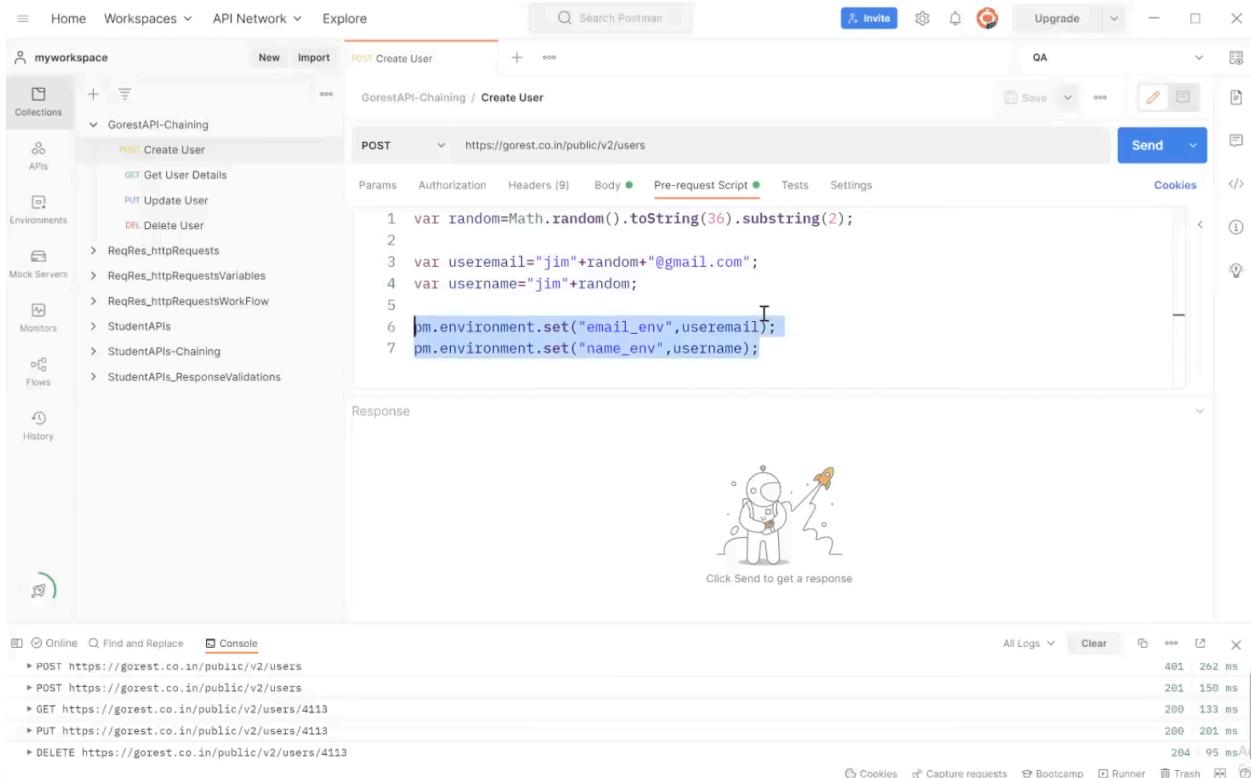
Ben bu konuyu çalışırken kendim test için Go Rest API'ın endpoint'lerini kullandım
notu okuyacak kişi oradan practice yapabilir işinize yarayabilir

Token: Erişim anahtarı gibi düşünülebilir

(Go Rest'e erişmek için token lazımdı hesap açılınca oluşuyor)



POST için gerekli olan variable'lar pre-request kısmında oluşturulur sonra onları enviroment variable'ın içine koyup bütün requestlerde çağrılabılırız:



The screenshot shows the Postman application interface. On the left, there's a sidebar with 'myworkspace' selected, containing sections for Collections, APIs, Environments, Mock Servers, Monitors, and Flows. A 'History' section is also present. The main workspace shows a collection named 'GorestAPI-Chaining' with a single 'Create User' POST request. The 'Pre-request Script' tab for this request contains the following JavaScript code:

```
1 var random=Math.random().toString(36).substring(2);
2
3 var useremail="jim"+random+"@gmail.com";
4 var username="jim"+random;
5
6 pm.environment.set("email_env",useremail);
7 pm.environment.set("name_env",username);
```

Below the script, there's a 'Response' section with a small cartoon character and a placeholder text 'Click Send to get a response'. At the bottom of the screen, the 'Console' tab is open, showing a list of API requests and their responses:

Request	Response Time
POST https://gorest.co.in/public/v2/users	401 262 ms
POST https://gorest.co.in/public/v2/users	261 150 ms
GET https://gorest.co.in/public/v2/users/4113	260 133 ms
PUT https://gorest.co.in/public/v2/users/4113	260 201 ms
DELETE https://gorest.co.in/public/v2/users/4113	264 95 ms

süslü parantezlerle post body'e variablelar eklenenir:

The screenshot shows the Postman application interface. On the left, the sidebar displays 'myworkspace' with various collections like 'GorestAPI-Chaining', 'ReqRes_httpRequests', and 'StudentAPIs'. The main workspace shows a POST request to 'https://gorest.co.in/public/v2/users'. The 'Body' tab is selected, showing a JSON payload with variables: "name": "{{name_env}}", "gender": "male", "email": "{{email_env}}", and "status": "inactive". Below the request, the 'Response' section features a cartoon illustration of a character with a rocket. At the bottom, the 'Logs' panel shows several API requests and their responses.

```

1 "name": "{{name_env}}",
2 "gender": "male",
3 "email": "{{email_env}}",
4 "status": "inactive"
    
```

Request	Response	Time
POST https://gorest.co.in/public/v2/users	201	262 ms
POST https://gorest.co.in/public/v2/users	201	159 ms
GET https://gorest.co.in/public/v2/users/4113	200	133 ms
PUT https://gorest.co.in/public/v2/users/4113	200	201 ms
DELETE https://gorest.co.in/public/v2/users/4113	204	95 ms

Küçük Not: Response değerinin bir variable'a konması Tests kısmında gerçekleştirilebilir çünkü Tests kısmı en son çalıştırılır bu bir önceki hiyerarşinin anlatıldığı notlarımmda yer almaktadır sadece hatırlatma için buraya tekrar koydum

The screenshot shows the Postman application interface. On the left, the sidebar displays the workspace structure under 'myworkspace'. A collection named 'GorestAPI-Chaining' is expanded, showing requests for 'Create User', 'Get User Details', 'Update User', and 'Delete User'. Below this, there are sections for 'ReqRes_httpRequests', 'ReqRes_httpRequestsVariables', 'ReqRes_httpRequestsWorkFlow', 'StudentAPIs', 'StudentAPIs-Chaining', and 'StudentAPIs_ResponseValidations'. The main central area shows a POST request titled 'Create User' to 'https://gorest.co.in/public/v2/users'. The 'Tests' tab contains a script:

```
1 var jsonData=JSON.parse(responseBody);
2 pm.environment.set("userid_env",jsonData.id);
```

The 'Body' tab shows the response JSON:

```
1 {
2   "id": 4131,
3   "name": "jim1cjhp1r72n7",
4   "email": "jim1cjhp1r72n7@gmail.com",
5   "gender": "male",
6   "status": "inactive"
7 }
```

The status bar at the bottom indicates the response was '201 Created' with a time of 159 ms and a size of 1.36 KB.

sonrasında bu `userid_env` Get requestin içinde kullanılıp comparison validation yapılır.

Aşağıda comparison validationun nasıl yapıldığı belirtilmiştir:

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Home, Workspaces, API Network, and Explore. Under 'myworkspace', there are collections like 'GorestAPI-Chaining' containing 'Create User', 'Get User Details', 'Update User', and 'Delete User'. Below these are environments, mock servers, monitors, flows, and history. The main workspace on the right shows a 'GorestAPI-Chaining / Get User Details' request. It's a GET request to https://gorest.co.in/public/v2/users/{{userid_env}}. The 'Tests' tab is selected, displaying the following JavaScript code:

```
1 //validating json fields in the response
2
3 pm.test("values of json fields", () =>{
4     var jsonData=pm.response.json();
5
6     pm.expect(jsonData.id).to.eql(pm.environment.get("userid_env"));
7     pm.expect(jsonData.email).to.eql(pm.environment.get("email_env"));
8     pm.expect(jsonData.name).to.eql(pm.environment.get("name_env"));
9
10 }
11 );
```

Below the code editor is a 'Response' section featuring a cartoon character holding a rocket. At the bottom of the interface, there's a 'Console' tab showing a list of recent requests and their responses:

- GET https://gorest.co.in/public/v2/users/4113
- PUT https://gorest.co.in/public/v2/users/4113
- DELETE https://gorest.co.in/public/v2/users/4113
- POST https://gorest.co.in/public/v2/users
- POST https://gorest.co.in/public/v2/users

At the very bottom, there are buttons for Cookies, Capture requests, Bootcamp, Runner, Trash, and a settings icon.

en son Delete kısmında variable'lar unset edilip CRUD işlemlerinin sonuna gelinmiştir:

The screenshot shows the Postman application interface. On the left, the sidebar displays 'myworkspace' with various collections like 'GorestAPI-Chaining', 'ReqRes_httpRequests', and 'StudentAPIs'. The main area shows a 'Delete User' request under 'GorestAPI-Chaining'. The 'Tests' tab contains a script:

```

1 pm.environment.unset("userid_env");
2 pm.environment.unset("email_env");
3 pm.environment.unset("name_env");
4

```

The 'Console' tab at the bottom shows the response body of the DELETE request, which includes headers and a timestamp. The 'Logs' tab shows a list of recent requests and their status codes.