# API Response Validations

## Response Validations

Aşağıda Response'ta validate edilecek değerler verilmiştir:

- Status code
- Headers
- Cookies
- Response Time
- Response Body

Assertion = Validation Point = kısaca yukardakileri validate yapmak için yazılan fonksiyonlar

PM - Bu fonksiyonları içeren JS kütüphanesi

# TESTING STATUS CODES

Response'dan gelen status code için test:

```
pm.test("Status code is 200", () => {
    pm.response.to.have.status(200);
});
```

birden fazla status code'u onaylamak için aşağıdaki gibi yazılmalıdır:

```
pm.test("Successful POST request", () => {
    pm.expect(pm.response.code).to.be.oneOf([201, 202]);
});
```

Status code'un textini kontrol ederken:

```
pm.test("Status code name has string", () => {
    pm.response.to.have.status("Created");
});
```

# Testing Headers

Content-Type Response içeriğinde “Header’ı geliyor mu? “ Kodu:

```
pm.test("Content-Type header is present", () => {
    pm.response.to.have.header("Content-Type");
});
```

Headerın içeriğini kontrol etmek için:

Aşağıdaki kod şunu kontrol eder Content-Type headerı olucak içeriği de verildiği gibi olcak mı diye bakar

farklıysa test FAIL

```
pm.expect(pm.response.headers.get("Content-Type"))
  .to.eql("application/json; charset=utf-8");
```

present = existence check

particular = value check

# Testing Cookies

‘Language’ isimli Cookie geliyor mu?:

```
pm.test("Cookie 'language' is present", () => {
    pm.expect(pm.cookies.has('language')).to.be.true;
});
```

‘Language’ isimli Cookie’nin içeriğine bakar:

```
pm.test("Cookie language has value 1", () => {
    pm.expect(pm.cookies.get('language')).to.eql('en-gb');
});
```

# Testing Response Time

Response süresinin bellow veya above olarak ayarlanıp test edilmesine yarıyor

```
pm.test("Response time is less than 200ms", () => {
    pm.expect(pm.response.responseTime).to.be.below(200);
});
```

# Response Body Test

Response’un parça parça test edilmesi:

Response:

```
{
  "id": 1,
  "name": "John",
  "location": "india",
  "phone": "1234567890",
  "courses": [
    "Java",
    "Selenium"
```

```
  ]
}
```

Type Testing:

```
const jsonData = pm.response.json();

pm.test("Test data type of the response", () => {
    pm.expect(jsonData).to.be.an("object");
    pm.expect(jsonData.name).to.be.a("string");
    pm.expect(jsonData.id).to.be.a("number");
    pm.expect(jsonData.courses).to.be.an("array");
});
```

```
pm.test("Test array properties", () => {
    // Java varmı yokmu ona bakar
    pm.expect(jsonData.courses).to.include("Java");

    // aşağıdakiler olmak zorunda ona bakar
    pm.expect(jsonData.courses)
        .to.have.members(["Java", "Selenium"]);
});
```

Json alanlarının değer testi:

```
pm.test("value of location field is India", () => {
    var jsonData = pm.response.json();

    pm.expect(jsonData.id).to.eql(1);
    pm.expect(jsonData.name).to.eql("John");
    pm.expect(jsonData.location).to.eql("india");
    pm.expect(jsonData.phone).to.eql("1234567890");
```

```
    pm.expect(jsonData.courses[0]).to.eql("Java");
    pm.expect(jsonData.courses[1]).to.eql("Selenium");
});
```

# Validating JSON Schema

Json Schema Nedir?

**JSON Schema**, bir JSON'un

👉 **nasıl görünmesi gerektiğini**

👉 **hangi alanlar olacak**

👉 **hangi tipte olacaklar**

👉 **hangileri zorunlu**

bunları **kural olarak tanımlayan** bir şemadır.

Json Schema:

```
var schema = {
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "id": {
      "type": "integer"
    },
    "name": {
      "type": "string"
    },
    "location": {
      "type": "string"
```

```
    },
    "phone": {
      "type": "string"
    },
    "courses": {
      "type": "array",
      "items": [
        {
          "type": "string"
        },
        {
          "type": "string"
        }
      ]
    }
  },
  "required": [
    "id"
  ]
};
```

Jsonu, Json Schema'ya çevirmek için bir çok online tool var kullanabilirsiniz örnek:

https://www.liquid-technologies.com/online-json-to-schema-converter

Json schema validation kodu:

```
pm.test('Schema is valid', function () {
    pm.expect(tv4.validate(jsonData, schema)).to.be.true;
});
```

Son not: Get Request diğer Requestlere göre daha çok testing içermelidir diğerleri headers cookies vs testlerini içerir ama get request response body içerdiği için

daha fazla data testi uygulanır.