**SNIPPETS**

**//Status kod dogrulama ---> Status Code: Code is 200**

pm.test("Durum Kodu = 200", function () {

pm.response.to.have.status(200);

});

**// Hız Dogrulama ---> Response time is less than 200ms**

pm.test("Response time is less than 200ms", function () { // hiz testi yapiyoruz 200 kismini degistirebiliriz

pm.expect(pm.response.responseTime).to.be.below(1000);

});

**// Body deki veriyi dogrulama ---> Response Body: Contains String**

pm.test("Body matches string", function () {

pm.expect(pm.response.text()).to.include("Köpek");

pm.expect(pm.response.text()).to.include("Pamuk");

pm.expect(pm.response.text()).to.include("Sibirya Kurdu");

});

**// Header bolumundeki basliklari dogrulama > Response headers: COntent-Type header Check**

pm.test("Content-Type is present", function () {

pm.response.to.have.header("Content-Type");

});

**// Status kod basliginin dogrulanmasi ---> Status code: Code name has string**

pm.test("Status code name has string", function () {

pm.response.to.have.status("OK");

});

**//Body de ki Key-Value degerlerinin eslestirilerek dogrulanmasi> Response body:JSON value check**

pm.test("Body deki Key-Value degerleri eslestirildi", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.id).to.eql(1909);

pm.expect(jsonData.name).to.eql("Pamuk");

pm.expect(jsonData.status).to.eql("available");

pm.expect(jsonData.category.id).to.eql(0);

pm.expect(jsonData.category.name).to.eql("kopek");

pm.expect(jsonData.category.id).to.eql(0);

pm.expect(jsonData.photoUrls[0]).to.eql("string"); // photoURL in 0. indexi "string" dedik. body den aliyoruz bu bilgileri

pm.expect(jsonData.tags[0].id).to.eql(0);

pm.expect(jsonData.tags[0].name).to.eql("Sibirya Kurdu"); // her suslu{} bir index demek.[] koyduk cunku "tags" dan sonra [] vardi

});

**//Header da ki Key-Value degerlerinin eslestirilerek dogrulanmasi > Body Check'in aynisi ama jSOn yazan yerleri header olarak degistiriyoruz asagida yazdigim gibi**

pm.test("Header da ki Key-Value degerleri eslestirildi", function () {

var headerData = pm.response.headers; // burada jSon() yaziyordu ama biz Header dakileri istiyoruz o yuzden jsonData yazan her yeri headerData diye degistirdik

pm.expect(headerData.get("Content-Type")).to.eql("application/json");

pm.expect(headerData.get("Server")).to.eql("Jetty(9.2.9.v20150224)");

});

**// Listenin uzunlugunu dogrulama**

pm.test("Sıfırdan çok, 500'den az mı?", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.length).to.be.greaterThan(0); // uzunluk icin value yerine length yazdik ve to.be.greaterThan ile degistirdik.JSon datanin uzunlugu 0 dan buyuk olsun dedik

pm.expect(jsonData.length).to.be.lessThan(500); // icerde 304 tane ve ve sinir rakam oarak 500 dedik.

});

**// Token alip Authenticetion yapmak --> Set collection variable seciyoruz**

pm.collectionVariables.set("token", pm.response.json().token); // bunu Collection icinde Variables kisminda gorebiliriz. O kisma bakmaliyiz.

**--------------------------------------------------------------------------------------------------**

**04.10.23 DERS NOTLARI**

**POST REQUEST OLUSTURMAK------->>>>> ( veri tabanına veri kaydettik bu şekilde)**

temel işlemlerin üç noktasına tıklayım add request dedik -> ismi degistirdk-> post sectik asagidan-> url yapistirdik swaggerdan aldigimiz -> sonra body-raw-json seciyoruz-> sonra swagger dokumanının oldugu yerden body yi kopyaladim geldim body altina yapistirdim.->body de degistirilecek yerleri degistirdik ve send dedik. yaptiklarimizi GET kismindan send deyip olmus mu diye kontrol ediyoruz, yani var olan bilgiyi cagiriyoruz

-- GET in icinde body olmayacak POST un icerisinde body olacak (yani eger veri tabanina veri girisi ya da degiskligi yapacaksak body kullanmaliyiz. eger GET gibi veri tabanina veri girisi yapmaycaksak body kullanmamiza gerek yok)

**PUT REQUEST OLUSTURMAK ------>>>>> (veri tabanindaki veriyi update ettik bu sekile)**

(post yerine put secilecek) yukaridaki islemler aynen yapilacak url yapistirilma yerine kadar.

put işleminde de body olmalı. cunku veri tabanına veri giricez yine. yani-> body-raw-json secimini yapmalıyız-> sonrasnda swagger dan body aldik geldik ppostmana yapistirdik-> id kismi postda olusturdugumuz id yi yazdik bunu degistirmedik, geri kalan degiskenleri istedigimiz gibi deistirdik-> send yaptik. -> yaptiklarimizi GET kismindan send deyip olmus mu diye kontrol ediyoruz, yani var olan bilgiyi cagiriyoruz

**DELETE REQUEST OLUSTURMAK ----->>>>>>**

(POST yerine DELETE seciyoruz) url yi gelip yapistirma ya kadar ayni -> url yapistirdiktan sonda send dedik asagida status: 404 not found oldu -> asagidan konsoldan json sectigimizde üc satirlik kod gelmeli. Eger bu kod gelimiyorsa vertabaninda bilgiyi 15 dkka tuttugunu hatirliuoruz ve gidip POST kismindan verimizi tekrar send yapiyoruz. DELETE ksmina gelip tekrar send dedigmizde bu sefer silme islamini yapiyor. uc satirlik kod geliyor goruyoruz.

**05.10.23 DERS NOTLARI**

{} -> jsona data, [] -> list demek bunlari anlamaliyiz

--> name-id-status gibi taglar key-value eslestirilmesi olarak dogrulanabilir

pm.expect(jsonData.category.id).to.eql(0); pm.expect(jsonData.category.name).to.eql("kopek");

--> yukaridaki sekilde category.name seklinde yazarak category altinda ki name vs gibi onlari da dogrulayabiliriz. her {} bir index demek. index 0 dan baslar.

--> pm.expect(jsonData.tags[0].id).to.eql(0); -> mesela burada tags altinda 0. index id imis ve degeri 0 mis bunu anliyoruz.

--> pm.expect(jsonData.tags[0].name).to.eql("Sibirya Kurdu"); -> mesela burada tags altinda 0. indexte name varmis bir de ve degeri Sibirya Kurdu imis bunu anliyoruz.

body deki key value degerleri kasilastirilirken pm.test ici söyle oluyorken;

var jsonData = pm.response.json();

pm.expect(jsonData.id).to.eql(1909);

headers da ki key value degerleri karilastirilirken pm.test ici soyle oluyor;

var headerData = pm.response.headers

pm.expect(headerData.get("Content-Type")).to.eql("application/json");

headerData dan sonra .get() demeli ve key kismini oraya yazmaliyiz

**06.10.23 DERS NOTLARI**

-> Status ile arama yaparken swagger a gidip GET taginin icinde findByStatus olanisecip tri iy out dedik. Sonrasinda asagida cikan, "available-pending-sold(buraya kendimizde ekleme yapabiliyormuşuz)" dan birini sectik.

-> sonrasinda asagi inip her zaman yaptigimiz gibi Request URL degil Curl olanin icindeki url yi alip Postman'a geldik.

-> sol panelin yukarisindaki Import'a tiklayip copyaladigimiz url yi cikan ekran yapistirdik ve bize kendiliginden GET olusturdu.

-> bu olusan Status ile GET olusturmadir. Bu sebeple soldaki menuye otomatik olarak eklemez. Eklemek icin -> Save diyoruz ve bizden isim degistirmemizi istiyor degistirip nereye kaydedilecegini seciyoruz ve boylece sol panele ekleniyor

--->>> bu olusan Status ile Arama GET sorgusunu Send yaptigimizda bize JSon olarak dondurmuyor sonucu List olarak donduruyor.

**07.10.23 DERS NOTLARI**

-> Postman de yeni bir collection olusturduk. adı: Contact List

-> Sonrasinda Add Request dedik ve <https://documenter.getpostman.com/view/4012288/TzK2bEa8#e388e723-b3e6-4a9b-99d5-609cbd6d4a4d> linkinden soldaki panelde contact a tikladik ve altinda acilan Get Contact List e tikladik. Acilan sayfadaki url yi aldik Postmanda olusturdugumuz GET Contact Lis'e e yapistirdik.

-> Send e tklayinca bizden token istedi ("Please authenticate." dedi konsol). ardindan <https://thinking-tester-contact-list.herokuapp.com/contactList> bu siteye gelip bi yere sag tik dedik "Application" u sectik-> soldan Cookiesin altindakini sectik ve tokeni aldik kopyaladik o bi dursun kenarda.

-> sonrasinda Postmana geldik ve Headers bolumune tikladik. sonrasinda <https://documenter.getpostman.com/view/4012288/TzK2bEa8#e388e723-b3e6-4a9b-99d5-609cbd6d4a4d> a gidip Headers basligi altindaki Key-Value lari alip Postman'de Headers altindaki kisma ekledik. Bearer {{token}} yazan yere yukarida aldigimiz tokeni ekledik. Susluleri kaldirip ekliyoruz token i ve sonrasinda Save ardindan Send diyoruz.

-> Konsolda [] bos bir List goruyoruz. Sebebi sitede her hangi bir veri girisi yapmadik.

-> Bu yuzden Contact List Documantion sitesinden aldigimiz verileri <https://thinking-tester-contact-list.herokuapp.com/contactList> sitesindeki Add a New Contact diyerek ekliyoruz sonrasinda submit diyoruz. Boylece siteye veri girişi yapmis oluyoruz.

-> Postman'a geri gelip tekrar Send yaptigimizda bu kez konsolda siteye girdigimiz verileri goruyoruz.

Token islemini otomatik olarak yapabilmek icin yaptigimiz islemler---->>>>

Once Postman'da Post Login User adinda bir request olusturduk ve Contact List Documentaion sitesine gelip User->Log In User altindaki BOdy-Raw-Json kismini alip copy paste yaptik Postman'de. Buradak mail ve password kisimlarini Contact List sitesinde giris yaparken kullandigimiz bilgilerle degistirdik ve save-send yaptik.

--> Sonrasinda ayni Post Login User requesti icindeki Test kismina gectik ve Snippets kismindan Set a colleciton variable i secerek bir sorgu olusturduk. buradaki ilk "" icine "token" yazdik ve virgulden sonraki ikinci "" kismina "pm.response.json().token" yazip save-send dedik.

-> Bu olusturdumuzu nerde gorduk? Postman'de olusturdugumuz Contact List collectionunun icindeki Variables kismina ekledigini gorduk. Ayni yerde Authoraiton kismina gelip Type kismini "Beraer token" sectik. Asagidan token karsina da "{{token}}" yazdik. Yni Collection duzeyinde degisken bir token olusturduk.

-> Get Contact List icinde Authoration kismini Inherit olarak secmemeiz lazim genelde zaten o secili olur. Parentten miras al demek bu.

**09.10.23 DERS NOTLARI**

--> Postman ile API testi yapacagiz bu derste. Postmanda senaryo yazimi, ornek testler, degisken ayarlamalari yapilacak, butun collectionlar yapilditan sonra rapor olustruma, takvime gore test kosma nasil olacak bunlari islicez bu derste.