Sign-Up API test scenarios

Post Method

URL:<https://fakestoreapi.com/auth/login>

1. **Valid sign up - Test login functionality with valid credentials.**

**Input -**

**Body -**

{“username”: “mor\_2314”,

“password”: “83r5^\_”

“firstname” : “abcd”,

“lastname” : “xyz”

}

**javascript -**

pm.test(“Status code is 200”,function(){

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

});

**Output -**

Status code is 200

**2.Invalid username - Test login functionality with a invalid username but valid password.**

**Input-**

**body -**

{ “username”: “asdf”,

“password” : “83r^5\_”

“firstname” : “abcd”,

“lastname” : “xyz”

}

**output-**

Status code - 401 Unauthorized

Response body - username and password is incorrect

**3.Invalid password - Testing login functionality with a valid username and invalid password.**

**Input-**

**body -**

{“username” : “mor\_2314”,

“password” : “Wrong password”

“firstname” : “abcd”,

“lastname” : “xyz”

}

**javascript -**

pm.test(“Status code is 401”,function(){

pm.response.to.have.status(401);

});

**Output -**

Status code 401 Unauthorized

Response body - username and password is incorrect

**4.Missing username - Test login functionality with a missing username.**

**Input-**

**body -**

{ “username” : “”,

“password” : “83r^5\_”

“firstname” : “abcd”,

“lastname” : “xyz”

}

**javascript -**

pm.test(“Status code is 400”,function(){

pm.response.to.have.status(400);

});

**output -**

Status code is 400

Response body - username is required

**5**. **Missing password - Test login functionality with a missing password.**

**Input-**

**body -**

{ “username” : “mor\_2314”,

“password” : “”

“firstname” : “abcd”,

“lastname” : “xyz”

}

**javascript -**

pm.test(“Status code is 400”,function(){

pm.response.to.have.status(400);

});

**output -**

Status code is 400

Response body - password is required

**6.Empty Request body - test login functionality with an empty request body**

**Input-**

**Body-**

{ “username” : “”,

“password” : “”

“firstname” : “”,

“lastname” : “”

}

**output -**

Status code is 400 Bad request

**7.Long username and password - Test login functionality with long username and password**

**Input -**

**Body** -

{ “username” : “asdfgdjdkofjnvmkcoehdofnssndbsdnbdasndbsmds”

“password” : “asjwheuwjmmdfkkfmfjeooepwoemfbvfdm”

“firstname” : “abcd”,

“lastname” : “xyz”

}

**javascript** -

pm.test(“Status code is 401”,function(){

pm.response.to.have.status(401);

});

**output** -

Status code is 401 unauthorized

Response body - username and password is incorrect

**8.Special characters - Test login functionality with special characters in username and password.**

input-

body-

{“username” -”$%%^&”,

“password”- “#$%^&”

“firstname” : “abcd”,

“lastname” : “xyz”

}

javascript -

pm.test(“Status code is 401”,function(){

pm.response.to.have.status(401);

});

output -

status code is 401 Unauthorized

Response body - username and password is incorrect

**9. Weak password**

input -

Body -

{“username”: “mor\_2314”,

“password”: “12345\_”

“firstname” : “abcd”,

“lastname” : “xyz”

}

javascript -

javascript -

pm.test(“Status code is 400”,function(){

pm.response.to.have.status(400);

});

Output -

Status - 400 bad request

**10. Duplicate username**

input -

Body -

{“username”: “exist123”,

“password”: “83r5^\_”,

“firstname” : “abcd”,

“lastname” : “xyz”

}

javascript -

pm.test(“Status code is 400”,function(){

pm.response.to.have.status(400);

});

Output -

Status - 400 bad request

username is incorrect

Get Method

Url - https://jsonplaceholder.typicode.com/users

1. **Get all users :**

https://jsonplaceholder.typicode.com/users

expected Response :

Status : 200 OK

Body : [ { "id": 1, "name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz", ... },

{“id” : 2….}]

1. **Get single user by ID**

<https://jsonplaceholder.typicode.com/users/1>

Expected response :

Status : 200 OK

Body : { "id": 1, "name": "Leanne Graham", "username": "Bret", "email": "Sincere@april.biz", ... }

1. **Get user with invalid query parameter**

<https://jsonplaceholder.typicode.com/users/id=xyz>

Expected Responses:

Status:200 Ok

Body: []

1. **Get users with valid query parameter**

<https://jsonplaceholder.typicode.com/users/id=2>

Expected Responses:

Status : 200 OK

Body: [ {

"id": 2,

"name": "Ervin Howell",

"username": "Antonette",

"email": "Shanna@melissa.tv",

"address": {

"street": "Victor Plains",

"suite": "Suite 879",

"city": "Wisokyburgh",

"zipcode": "90566-7771",

"geo": {

"lat": "-43.9509",

"lng": "-34.4618"

}

},

"phone": "010-692-6593 x09125",

"website": "anastasia.net",

"company": {

"name": "Deckow-Crist",

"catchPhrase": "Proactive didactic contingency",

"bs": "synergize scalable supply-chains"

}

}

]

1. **Get user with invalid user**

<https://jsonplaceholder.typicode.com/users/100>

Expected Responses:

Status : 404 Not Found

Body: []

1. **Get users with invalid multiple query**

[https://jsonplaceholder.typicode.com/users?id=100 & username=dfg](https://jsonplaceholder.typicode.com/users?id=100&username=dfg)

Expected Responses:

Status : 200 OK

Body: []

1. **Get users with valid multiple query**

<https://jsonplaceholder.typicode.com/users?id=1&username=Bret>

Expected Responses

Status code: 200 OK

Body:[ { "id": 1, "name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz", ... },

1. **Get user with checking response time**

<https://jsonplaceholder.typicode.com/users>

Input :

pm.test(“status code is 200” ,function[]{

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

});

Expected responses:

status: 200 OK

1. **Get user with no query parameters**

https;//[jsonplaceholder.typicode.com/users](http://jsonplaceholder.typicode.com/users)

Expected Response:

Status: 200 OK

Body : Body : { "id": 1, "name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz",

... }

1. **Get users with valid name**

<https://jsonplaceholder.typicode.com/users?name=LeanneGraham>

Expected Responses:

Status : 200 OK

Body : { "id": 1,

"name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz",

... }

1. **Get users with invalid name**

<https://jsonplaceholder.typicode.com/users?name=qwer>

Expected Responses:

Status : 200 OK

Body : []

PUT Method

<https://jsonplaceholder.typicode.com/users>

1. **Update user with valid data**

URL - <https://jsonplaceholder.typicode.com/user/1>

Input -

{ “id” :1,

“name” : “xyz”

}

Expected output -

Status- 200 OK

Body - { “id” :1,

“name” : “xyz”

}

1. **Update user with invalid data**

URL - <https://jsonplaceholder.typiocde.com/users/100>

input -

{

“id” : 100

“name” : “abc”

}

output -

Status - 404 Not found

1. **Update user with missing fields**

URL - <https://jsonplaceholder.typicode.com/user/1>

Input - { “id” :1}

expected output -

status - 200 OK

Body - {

“id” : 1 }

1. **Update user with Empty Body**

URL - <https://jsonplaceholder.typicode.com/user/1>

input - {}

Expected output-

Status - 200 OK

Body - {}

1. **Update user with Extra data field**

URL - <https://jsonplaceholder.typicode.com/user/1>

input - {

“id” : 2,

“name” : “xyz”,

“ email” : “email@.com”,

“age” :18}

Expected output -

Status - 200 OK

body -

{“id” : 2,

“name” : “xyz”,

“ email” : “email@.com”,

“age” :18}

1. **update user without name**

URL - <https://jsonplaceholder.typicode.com/user/1>

input - {

“id” : 2,

“ email” : “email@.com”,

“age” :18}

Expected output -

Status - 200 OK

Body - {

“id” : 2,

“ email” : “email@.com”,

“age” :18}

1. **update user with empty values**

URL - <https://jsonplaceholder.typicode.com/user/1>

input - { “ id” : 1,

“name” : “”,

“age” : “”

}

Expected output -

Status code - 200 OK

body - { “ id” : 1,

“name” : “”,

“age” : “”

}

1. **Update user with large values**

URL - <https://jsonplaceholder.typicode.com/user/1>

Input - {

“id” : 1,

“name “ : “ name with a long value qwertyuiop”

“age”: 2355653232

}

Expected output:

Status - 200OK

Body - “id” : 1,

“name “ : “ name with a long value qwertyuiop”

“age”: 2355653232

}

1. **update user with special characters**

URL - <https://jsonplaceholder.typicode.com/user/1>

input - {

“id”: 1,

“name”: “abc@#$”,

“age” : 18

}

Expected output -

Status code - 200 OK

body - {

“id”: 1,

“name”: “abc@#$”,

“age” : 18

}

1. **update user with invalid endpoints**

URL - <https://jsonplaceholder.typicode.com/user/1>

Input -

{

“id” : 1,

“name” : “qwer”,

“age” : 18,

}

Expected output -

status - 404 Not found

body - {}

**Patch Method**

URL - <https://jsonplaceholder.typicode.com/user/1>

1. **update name -**

input - {

“name” : “asdfgg” }

expected output -

Status - 200 OK

body - { "id": 1,

"name": "asdfgg",

"username": "Bret",

"email": "Sincere@april.biz",

... }

1. **update email -**

input -

{ “email” : “zxv@.com”}

Expected output -

Status -200 OK

Body - { "id": 1,

"name": "Leanne Graham",

"username": "Bret",

"email": "zxv@.com",

... }

1. **update with invalid user**

input - {

“username” : “asdf”}

Expected output -

Status - 200 OK

body - {}

1. **update user with invalid data type**

input - {

“email” - “ 12345”

}

expected output -

Status - 200 OK

body - { "id": 1,

"name": "Leann Graham",

"username": "Bret",

"email": "12345",

... }

1. **update user with empty body**

input - {}

Expected output -

Status - 200 OK

body - { "id": 1,

"name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz", ... },

1. **update user with null values**

input -

{ “email” : “” }

expected output -

Status - 200 OK

body - { "id": 1,

"name": "Leanne Graham",

"username": "Bret",

"email": " ”, ... },

**7.update user with special characters**

input - {

“username” : “awee@$%^”

}

Expected output -

Status - 200 OK

Body - { "id": 1,

"name": "Leanne Graham",

"username": "awee@$%^",

"email": " ”, ... },

**8. update user with large values**

input - {

“username” : “username with very long values”

}

expected output -

Status - 200 OK

body - { "id": 1,

"name": "Leanne Graham",

"username": "username with very long values",

"email": " ”, ... },

**9. update user phone number**

input -

{ “phone” : “123456789”

}

Expected output -

status- 200 OK

body - { "id": 1,

"name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz",

“phone” : 123456789

... },

**10. update multiple fields**

input -

{

“name” : “divya”,

“username” : “qwer”,

“email” : “qwet@.com”

}

expected output -

Status - 200 OK

body - {

“name” : “divya”,

“username” : “qwer”,

“email” : “qwet@.com”

“phone” :123456789

}