

# Postman API Testing

## Signin

input:

```
{  
  
  "username": "admin_user",  
  "password": "adminpass"  
}
```

output:

```
{ "lastName": "User", "country": "AnyCountry", "city": "Anytown", "postalCode": "12345", "token": "642c08fa-0491-11ef-8acf-544e839a0c04", "firstName": "Admin", "streetAddress": "123 Main St", "permissions": "Create,Read,Update,Delete", "contactNumber": "1234567890", "state": "AnyState", "userRole": "Admin", "email": "admin@example.com", "username": "admin_user" }
```

url: <http://localhost:8080/loginapi/signin>

if Username and Password is correct

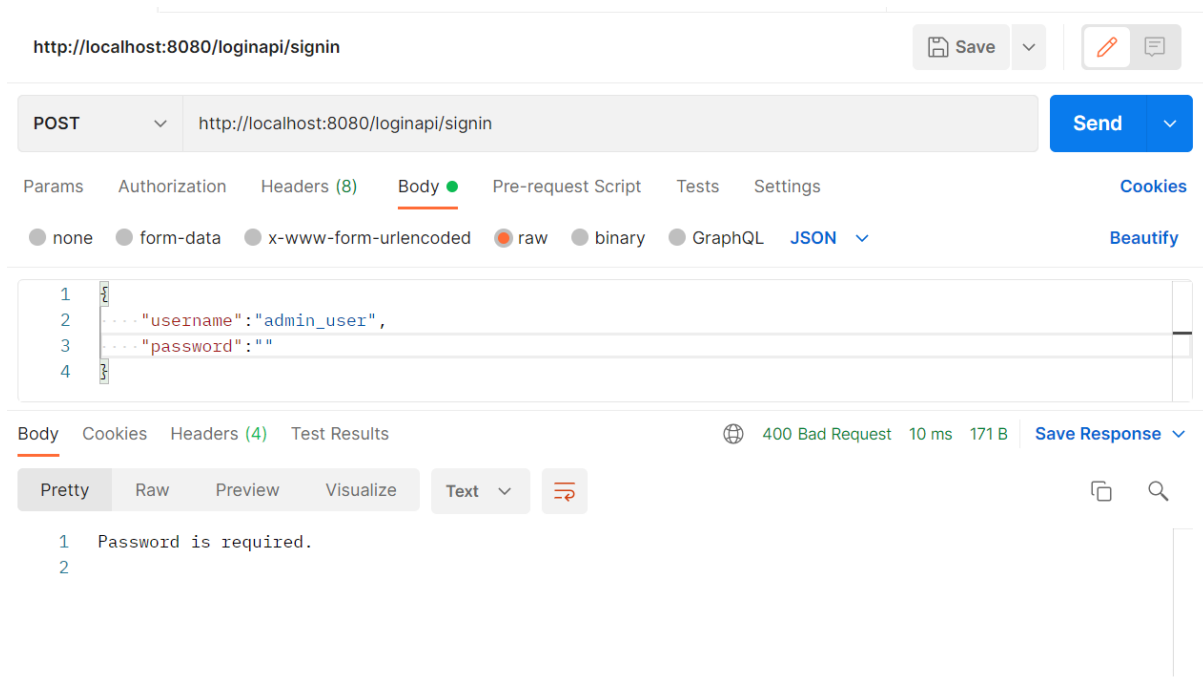
The screenshot shows the Postman interface for a POST request to `http://localhost:8080/loginapi/signin`. The request body is a JSON object with the following fields: `username` (admin\_user), `password` (adminpass). The response is a 200 OK status with a response time of 453 ms and a body size of 508 B. The response body is a JSON object with the following fields: `lastName` (User), `country` (AnyCountry), `city` (Anytown), `postalCode` (12345), `token` (642c08fa-0491-11ef-8acf-544e839a0c04), `firstName` (Admin), `streetAddress` (123 Main St), `permissions` (Create,Read,Update,Delete), `contactNumber` (1234567890), `state` (AnyState), `userRole` (Admin), `email` (admin@example.com), and `username` (admin\_user).

```
1 {  
2   "username": "admin_user",  
3   "password": "adminpass"  
4 }
```

Body Cookies Headers (5) Test Results 200 OK 453 ms 508 B Save Response

```
1 {  
  "lastName": "User", "country": "AnyCountry", "city": "Anytown", "postalCode": "12345",  
  "token": "642c08fa-0491-11ef-8acf-544e839a0c04", "firstName": "Admin", "streetAddress": "123 Main St",  
  "permissions": "Create,Read,Update,Delete", "contactNumber": "1234567890", "state": "AnyState",  
  "userRole": "Admin", "email": "admin@example.com", "username": "admin_user"  
}
```

## If any of the details is missing



http://localhost:8080/loginapi/signin

POST http://localhost:8080/loginapi/signin

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON Beautify

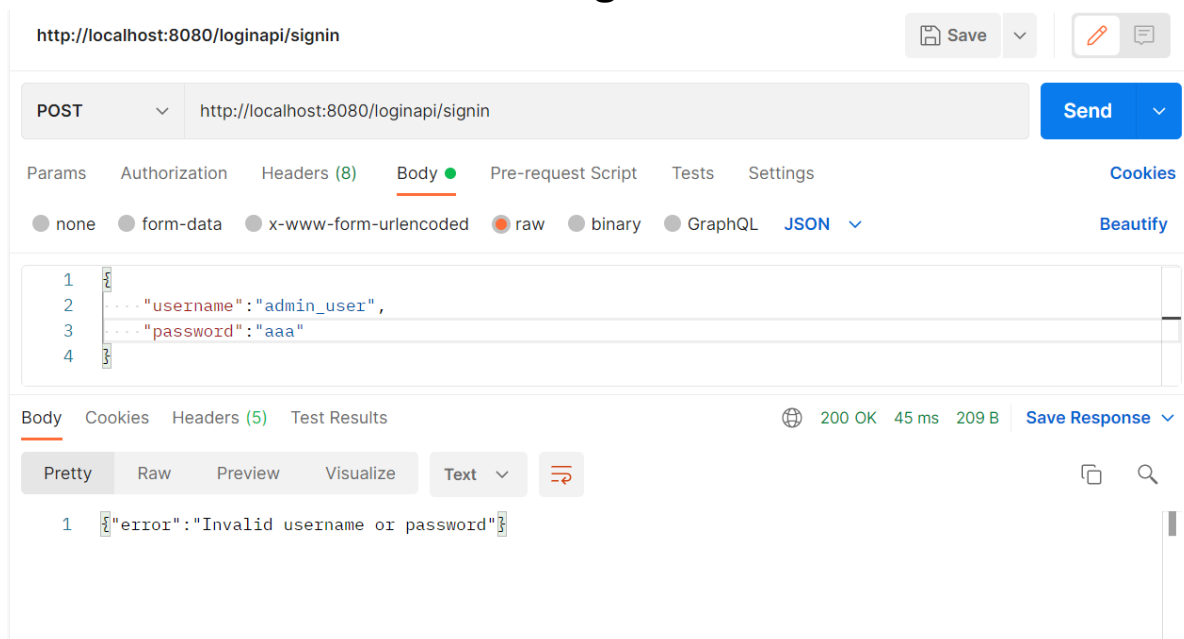
```
1 {
2   "username": "admin_user",
3   "password": ""
4 }
```

Body Cookies Headers (4) Test Results 400 Bad Request 10 ms 171 B Save Response

Pretty Raw Preview Visualize Text

```
1 Password is required.
2
```

## If Password/Username is wrong



http://localhost:8080/loginapi/signin

POST http://localhost:8080/loginapi/signin

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON Beautify

```
1 {
2   "username": "admin_user",
3   "password": "aaa"
4 }
```

Body Cookies Headers (5) Test Results 200 OK 45 ms 209 B Save Response

Pretty Raw Preview Visualize Text

```
1 {"error": "Invalid username or password"}
```

## //List Ingredient API

**Input: Not Required**

**Output:**

```
{"ingredients":[{"name":"Flour","left_unit":20,"total_unit":50,"id":1}, {"name":"Sugar","left_unit":15,"total_unit":30,"id":2}, {"name":"Salt","left_unit":10,"total_unit":20,"id":3}, {"name":"Milk","left_unit":20,"total_unit":40,"id":4}, {"name":"Eggs","left_unit":30,"total_unit":60,"id":5}]}
```

**Url:** http://localhost:8080/loginapi/ingredients

http://localhost:8080/loginapi/ingredients

GET http://localhost:8080/loginapi/ingredients Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Type: API Key

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Key: TOKEN

Value: 313c06286a7544af832ab1ace10e868c

Add to Header

Body Cookies Headers (5) Test Results 200 OK 104 ms 459 B Save Response

Pretty Raw Preview Visualize Text

```
1 [{"ingredients":[{"name":"Flour","left_unit":20,"total_unit":50,"id":1}, {"name":"Sugar","left_unit":15,"total_unit":30,"id":2}, {"name":"Salt","left_unit":10,"total_unit":20,"id":3}, {"name":"Milk","left_unit":20,"total_unit":40,"id":4}, {"name":"Eggs","left_unit":30,"total_unit":60,"id":5}]}
```

## If token is invalid then

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8080/loginapi/ingredients`
- Method:** GET
- Authorization:** API Key (TOKEN) with value `313c06286a7544af832ab1ace10e868`
- Response:** 400 Bad Request (37 ms, 176 B). The response body is `{"error": "TOKEN_NOT_VALID"}`.

## //Update User Status API

**Url:** `http://localhost:8080/loginapi/userstatus/{userId}/{status}`

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8080/loginapi/userstatus/6/active`
- Method:** POST
- Body:** none
- Response:** 200 OK (65 ms, 202 B). The response body is `User status updated successfully.`

If user don't have permission then user wont be able to update status only specific user whom update role has been assigned can update the status

The screenshot displays a REST client interface with the following details:

- URL:** `http://localhost:8080/loginapi/userstatus/6/active`
- Method:** POST
- Authorization:** API Key (TOKEN) with Value `642c08fa-0491-11ef-8acf-544e839a0c`
- Response:** 200 OK, 70 ms, 237 B. The response body is: `1 You Don't have peirmission to Update User Status Please contact Admin`

## //Update Ingredient API

This is also a private api accessible to only specific user

Input: {

```
"totalUnit": "50",  
"leftUnit": "20",  
"id": "1"
```

}

output: {"success": true, "message": "Ingredient with ID 1 updated successfully."}

Url: http://localhost:8080/loginapi/updateingredients

The screenshot displays a REST client interface with the following details:

- URL:** http://localhost:8080/loginapi/updateingredients
- Method:** POST
- Body (JSON):**

```
{  
  "totalUnit": "50",  
  "leftUnit": "20",  
  "id": "1"  
}
```
- Response:** 200 OK, 83 ms, 240 B. The response body is: 

```
{"success": true, "message": "Ingredient with ID 1 updated successfully."}
```

//Register User API

Input: {

```
"userType": "User",  
"username": "user1",  
"password": "user123@",  
"email": "xyz@gmail.com",  
"firstName": "Test",  
"lastName": "User",  
"contactNumber": "1234567890",  
"streetAddress": "street",  
"city": "city",  
"state": "state",  
"country": "country",  
"postalCode": "231320"
```

}

output: User Added Successfully

Url: <http://localhost:8080/loginapi/register>

http://localhost:8080/loginapi/register

POST http://localhost:8080/loginapi/register Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2   "userType": "User",
3   "username": "user1",
4   "password": "user123@",
5   "email": "xyz@gmail.com",
6   "firstName": "Test",
7   "lastName": "Heor"
```

Body Cookies Headers (5) Test Results 200 OK 123 ms 192 B Save Response

Pretty Raw Preview Visualize Text

```
1 User Added Successfully
```

If user already exist then

http://localhost:8080/loginapi/register

POST http://localhost:8080/loginapi/register Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2   "userType": "User",
3   "username": "user1",
4   "password": "user123@",
5   "email": "xyz@gmail.com",
6   "firstName": "Test",
7   "lastName": "Heor"
```

Body Cookies Headers (5) Test Results 200 OK 84 ms 187 B Save Response

Pretty Raw Preview Visualize Text

```
1 User Already Exist
```



if any field is missing then

http://localhost:8080/loginapi/register

Save

POST

http://localhost:8080/loginapi/register

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

Text

```
1 {
2   "userType": "",
3   "username": "",
4   "password": "user123@",
5   "email": "xyz@gmail.com",
6   "firstName": "Test",
7   "lastName": "User"
```

Body

Cookies

Headers (4)

Test Results

400 Bad Request

41 ms

194 B

Save Response

Pretty

Raw

Preview

Visualize

Text

```
1 Username is required.
2 User type is required.
3
```