**Blogging Application API Test Cases**

**Test Case 1: Register a new user:**

API endpoint: <https://api.realworld.io/api/users>

Request Payload:

{

  "user": {

    "username": "basanta",

    "email": "basanta.roy@gmail.com",

    "password": "testpassword"

  }

}

Expected Response:

* HTTP status code: ‘200 OK’
* Response Payload:

{

    "user": {

        "email": "basanta.roy@gmail.com",

        "username": "basanta",

        "bio": **null**,

        "image": "https://api.realworld.io/images/smiley-cyrus.jpeg",

        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6ImJhc2F udGEucm95QGdtYWlsLmNvbSIsInVzZXJuYW1lIjoiYmFzYW50YSIsImlhdCI6MTY3NjU1NjQ5NywiZXhwIjoxNjgxNzQwNDk3fQ.tz\_Ex6F2YC6IcxMwP3mYZ5gIgmn9H4djds3iB51dySw"

    }

}

Test Steps:

1. Send a POST request to the <https://api.realworld.io/api/users> endpoint with the above request payload.
2. Verify that the HTTP status code of the response is 200 OK.
3. Verify that the response payload contains a user object with the same username and email as the request payload.
4. Verify that the token field in the user object is a valid JWT token.

**Test Case 2: Login through newly created User:**

API endpoint: <https://api.realworld.io/api/users/login>

Request Payload:

{

  "user": {

    "email": "basanta.roy@gmail.com",

    "password": "testpassword"

  }

}

Expected Response:

* HTTP status code: ‘200 OK’
* Response payload:

{

    "user": {

        "email": "basanta.roy@gmail.com",

        "username": "basanta",

        "bio": **null**,

        "image": "https://api.realworld.io/images/smiley-cyrus.jpeg",

        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6ImJhc2FudGEucm95QGdtYWlsLmNvbSIsInVzZXJuYW1lIjoiYmFzYW50YSIsImlhdCI6MTY3NjU1NzE2NCwiZXhwIjoxNjgxNzQxMTY0fQ.y7sMBC1BykhCVVhX1dzsnrb7-wS4ACg-MVx8JHkBSpo"

    }

}

Test Steps:

1. Send a POST request to the API endpoint <https://api.realworld.io/api/users/login> with email and password fields in the request body.
2. Verify the response status code is 200 OK.
3. Verify the response contains details user login credentials with JWT token.

**Test Case 3: Get most recent articles globally:**

API endpoint: <https://api.realworld.io/api/articles>

Test Steps:

1. Send a GET request to the API endpoint <https://api.realworld.io/api/articles> .
2. Add the Header: Content-type with value application/json.
3. Add the authorization token generated while logging into the system.
4. Send the request and verify that the response status code is 200 OK.
5. Verify that the response body contains an array of articles.
6. Verify that expected response should be as the following:

{

    "articles": [

        {

            "slug": "City-139410",

            "title": "City",

            "description": "This is a updated article for city called Biratnagar.",

            "body": "Today I am Updating my existing article through using postman hope it goes fine",

            "tagList": [

                "test",

                "api"

            ],

            "createdAt": "2023-02-15T15:31:52.008Z",

            "updatedAt": "2023-02-16T06:17:37.295Z",

            "favorited": **false**,

            "favoritesCount": 0,

            "author": {

                "username": "basanta",

                "bio": **null**,

                "image": "https://api.realworld.io/images/smiley-cyrus.jpeg",

                "following": **false**

            }

        },

**Test Case 4: Create an Article:**

API endpoint: <https://api.realworld.io/api/articles>

Request Payload:

{

    "article": {

        "title": "World",

        "description": "API test for Deleteing an World",

        "body": "I am testing new API test for Deleteing an article let's give a try is it working or not?",

        "tagList": ["test", "api"]

    }

}

Expected Response:

* HTTP status code: ‘200 OK’
* Response payload:

{

    "article": {

        "slug": "World-139410",

        "title": "World",

        "description": "API test for creating a World",

        "body": "I am testing new API test for creating an article let's give a try is it working or not?",

        "tagList": [

            "test",

            "api"

        ],

        "createdAt": "2023-02-16T14:48:44.053Z",

        "updatedAt": "2023-02-16T14:48:44.053Z",

        "favorited": **false**,

        "favoritesCount": 0,

        "author": {

            "username": "basanta",

            "bio": **null**,

            "image": "https://api.realworld.io/images/smiley-cyrus.jpeg",

            "following": **false**

        }

    }

}

Test Steps:

1. Send a POST request to the API endpoint <https://api.realworld.io/api/articles>.
2. Add the Header: Content-type with value application/json.
3. Add the authorization token generated while logging into the system.
4. Send the request and verify that the response status code is 200 OK.

**Test Case 5: Get an articles:**

API endpoint: <https://api.realworld.io/api/articles>

Test Steps:

1. Send a GET request to the API endpoint <https://api.realworld.io/api/articles> .
2. Add the Header: Content-type with value application/json.
3. Add the authorization token generated while logging into the system.
4. Add request payload in Test.

pm.**test**("Status code is 200", **function** () {

    pm.response.to.have.status(200);

});

pm.**test**("Response is JSON", **function** () {

    pm.response.to.be.json;

});

pm.**test**("Response has articles", **function** () {

    var jsonData **=** pm.response.json();

    pm.expect(jsonData.articles).to.not.be.empty;

});

1. Send the request and verify that the response status code is 200 OK.
2. Verify that the response body contains an array of articles.
3. Verify that expected response should be as the following:

{

    "articles": [

        {

            "slug": "City-139410",

            "title": "City",

            "description": "This is a updated article for city called Biratnagar.",

            "body": "Today I am Updating my existing article through using postman hope it goes fine",

            "tagList": [

                "test",

                "api"

            ],

            "createdAt": "2023-02-15T15:31:52.008Z",

            "updatedAt": "2023-02-16T06:17:37.295Z",

            "favorited": **false**,

            "favoritesCount": 0,

            "author": {

                "username": "basanta",

                "bio": **null**,

                "image": "https://api.realworld.io/images/smiley-cyrus.jpeg",

                "following": **false**

            }

        },

**Test Case 5: Update an articles:**

API endpoint: : <https://api.realworld.io/api/articles/World-139410>

Where Slug = World-139410 is article name which we are going to update

Request Payload:

{

    "article": {

        "title": "World",

        "description": "This is a updated article for city called Biratnagar.",

        "body": "Today I am Updating my existing article through using postman hope it goes fine",

        "tagList": ["test", "api"]

    }

}

Expected Response:

* HTTP status code: ‘200 OK’
* Response payload:

{

    "article": {

        "slug": "World-139410",

        "title": "World",

        "description": "This is a updated article for city called Biratnagar.",

        "body": "Today I am Updating my existing article through using postman hope it goes fine",

        "tagList": [

            "test",

            "api"

        ],

        "createdAt": "2023-02-16T14:48:44.053Z",

        "updatedAt": "2023-02-16T15:26:29.878Z",

        "favorited": **false**,

        "favoritesCount": 0,

        "author": {

            "username": "basanta",

            "bio": **null**,

            "image": "https://api.realworld.io/images/smiley-cyrus.jpeg",

            "following": **false**

        }

    }

}

Test Steps:

1. Send a PUT request to the API endpoint <https://api.realworld.io/api/articles/World-139410>.
2. Add the Header: Content-type with value application/json.
3. Add the authorization token generated while logging into the system.
4. Send the request and verify that the response status code is 200 OK.

**Test Case 6: Delete an articles:**

API endpoint: : <https://api.realworld.io/api/articles/World-139410>

Test Steps:

1. Send a DELETE request to the API endpoint<https://api.realworld.io/api/articles/World-139410> .
2. Add the Header: Content-type with value application/json.
3. Add the authorization token generated while logging into the system.
4. Send the request and verify that the response status code is 204.
5. As there is no content to display so the status code is shown 204.