

1. Get Post (Unauthenticated - HTTPS) <https://127.0.0.1:8000/api/posts/>

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

- Title:** GET 1. Get Post (Unauthenticated)
- Method:** GET
- URL:** https://127.0.0.1:8000/api/posts/
- Headers:** Authorization, Headers (6), Body, Scripts, Settings
- Query Params:** A table with one row:

Key	Value	Description
Key	Value	Description
- Body:** JSON (empty)
- Response Status:** 200 OK
- Response Headers:** 28 ms, 421 B
- Response Content:** An empty JSON array: []

2. Get Users (Unauthenticated - HTTPS) https://127.0.0.1:8000/api/users/

The screenshot shows the Postman application interface. At the top, there's a header bar with tabs for "Security (HTTPS)", "GET 2. Get Users (Unauthenticated)", and a "+" button. To the right are "Save", "Share", and "No environment" buttons.

The main workspace shows a "GET" request to "https://127.0.0.1:8000/api/users/". Below the URL, there are tabs for "Docs", "Params", "Authorization", "Headers (6)", "Body", "Scripts", and "Settings". The "Params" tab is selected, showing a table for "Query Params" with one row: "Key" (Value) and "Value" (Description).

At the bottom, there are tabs for "Body", "Cookies", "Headers (12)", and "Test Results". The "Body" tab is selected, showing the response content. The response status is "200 OK" with a timestamp of "14 ms" and a size of "524 B". The response body is displayed as JSON:

```
1 [  
2 {  
3   "id": 1,  
4   "username": "testadmin",  
5   "email": "test@example.com",  
6   "created_at": "2026-02-05T10:13:49.062009Z"  
7 }  
8 ]
```

3. Obtain JWT Token (HTTPS) <https://127.0.0.1:8000/api/token/>

The screenshot shows the Postman interface for a POST request to <https://127.0.0.1:8000/api/token/>. The request body is raw JSON with the following content:

```
1 {
2     "username": "testadmin",
3     "password": "admin1234!"
4 }
```

The response status is 200 OK, with a response time of 63 ms and a response size of 891 B. The response body is a JSON object containing two keys: "refresh" and "access".

```
1 {
2     "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJ0b2tlb190eXBlIjoicmVmcmVzaC1sImV4cCI6MTc3MDM3NDczOCwiaWF0IjoxNzcmMjg4MzM4LCJqdGkiOiI5YWY50GZkMwYzMmM0MDNhOTg2NGM4MThiNmIwOTA2NS  
IsInVzZXJfaWQiOixin0.1N58q7-dq90y9alqH7LDcJ_40N1jm1XM1CY5jd3xUzY",
3     "access": "eyJhbGciOiJIUzI1NiisInR5cCI6IkpXVCJ9.  
eyJ0b2tlb190eXBlIjoiyWNnjZXNzIiwiZXhwIjoxNzcmMjkxOTM4LCJpYXQiOjE3NzAyODgzMzsImp0aSI6ImQ4MzQ1NTNjYWJmZDQzMWI4ZjNmZGVhYmMxNzY2ODY0Ii  
widXNlc19pZCIjEifQ.HIO6rvsYHB89Pcn3Lr7me3-btwtog2CYj8yliiu_y9w"
4 }
```

4. Create User (Authenticated - HTTPS) <https://127.0.0.1:8000/api/users/>

The screenshot shows the Postman application interface. At the top, it displays "POST 4. Create User (Auther)" and "No environment". Below the header, the URL "https://127.0.0.1:8000/api/users/" is entered in the "POST" method field. The "Body" tab is selected, showing the raw JSON payload:

```
1 {
2     "username": "testuser1",
3     "email": "testuser1@example.com",
4     "password": "securepassword123"
5 }
```

Below the body, the response section shows a green "201 Created" status bar with "58 ms" and "524 B" metrics. The response body is displayed as JSON:

```
1 {
2     "id": 2,
3     "username": "testuser1",
4     "email": "testuser1@example.com",
5     "created_at": "2026-02-05T10:49:53.722266Z"
6 }
```

5. Create Post (Authenticated by Author - HTTPS) <https://127.0.0.1:8000/api/posts/>

The screenshot shows the Postman application interface. At the top, it displays "Security (HTTPS)" and "POST 5. Create Post (Author)". The URL in the header is "https://127.0.0.1:8000/api/posts/". The main area shows a POST request with the URL "https://127.0.0.1:8000/api/posts/". The "Body" tab is selected, showing the JSON payload:

```
1 {
2   "content": "My second authenticated post!"
3 }
```

Below the body, the "Test Results" section shows a successful response:

201 Created 24 ms 541 B Save Response

The response body is displayed as:

```
{ } JSON ▾
```

```
1 {
2   "id": 1,
3   "content": "My second authenticated post!",
4   "author_username": "testadmin",
5   "created_at": "2026-02-05T10:52:05.839454Z"
6 }
```

6. Update Post (Authenticated by Author - HTTPS) <https://127.0.0.1:8000/api/posts/2/>

The screenshot shows the Postman application interface. At the top, it displays the URL `PUT 6.. Update Post (Authenticated by Author) https://127.0.0.1:8000/api/posts/1/`. The main area shows a JSON payload being sent:

```
1 {  
2   "content": "Updated content for my second secure post"  
3 }
```

The response section shows a successful `200 OK` status with a response body containing the updated post details:

```
1 {  
2   "id": 1,  
3   "content": "Updated content for my second secure post",  
4   "author_username": "testadmin",  
5   "created_at": "2026-02-06T05:52:05.839454Z"  
6 }
```

7. Delete Post (Authenticated by Author - HTTPS) <https://127.0.0.1:8000/api/posts/2/>

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

- Header Bar:** Security (HTTPS) | DEL 7. Delete Post (Authenticated •) | + | No environment
- Request URL:** https://127.0.0.1:8000/api/posts/1/?
- Method:** DELETE
- Body:** Raw (selected), showing the number "1".
- Response Status:** 204 No Content
- Response Headers:** 14 ms, 401 B
- Buttons:** Save, Share, Send

8. Create Regular User (Authenticated - HTTPS) <https://127.0.0.1:8000/api/users/>

The screenshot shows the Postman interface for a POST request to create a regular user. The request URL is <https://127.0.0.1:8000/api/users/>. The request body contains the following JSON:

```
1 {
2     "username": "regularuser",
3     "email": "regular@example.com",
4     "password": "regularpassword"
5 }
```

The response status is 201 Created, with a response time of 12 ms and a response size of 524 B. The response body is:

```
1 {
2     "id": 4,
3     "username": "regularuser",
4     "email": "regular@example.com",
5     "created_at": "2026-02-05T11:03:42.922234Z"
6 }
```

9. Obtain JWT Token (Regular User - HTTPS) <https://127.0.0.1:8000/api/token/>

The screenshot shows the Postman interface with the following details:

- Request Method:** POST
- Request URL:** https://127.0.0.1:8000/api/token/
- Body Content:**

```
1 {
2     "username": "regularuser2",
3     "password": "regularpassword2"
4 }
```
- Response Status:** 200 OK
- Response Time:** 111 ms
- Response Size:** 891 B
- Response Body (JSON):**

```
1 {
2     "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
3         eyJ0b2tlbl9oeXBlijoicmVmcmVzaC1sImV4cI6Mtc3MDM3NjIxMSwiaWF0IjoxNzcwMjg50DExLCJqdGkiOiI50ThhODAwNmJkMzY0YTvhOGNhMzI40TQ4Yzc4ZmM0Yy
4             IsInvzXJfawQioii1In0.I1jQ4ML2WBTL2kxe7PXFtWHzdbiUjk8rTqhKzJBMs8",
5     "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
6         eyJ0b2tlbl9oeXBlijoicmVmcmVzaC1sImV4cI6Mtc3MDM3NjIxMSwiaWF0IjoxNzcwMjg50DExLCJqdGkiOiI50ThhODAwNmJkMzY0YTvhOGNhMzI40TQ4Yzc4ZmM0Yy
7             widXNlcl9pZCI6IjUiifQ.Jn6ysrw0TstbzDjvpkU-6w7vwewm2ktPIF9tjuxfPvA"
```

10. Regular User Create Post (HTTPS) <https://127.0.0.1:8000/api/posts/>

The screenshot shows the Postman application interface. At the top, there's a header bar with tabs for 'Security (HTTPS)', 'POST 10. Regular User Creat •', and a '+' button. To the right of the header are dropdowns for 'No environment' and a 'Save' button. Below the header, the main workspace has a 'POST' method selected, pointing to the URL 'https://127.0.0.1:8000/api/posts/'. The 'Body' tab is active, showing a raw JSON payload:

```
1 {  
2   "content": "Post by regular user 2"  
3 }
```

Below the body, the 'Headers' tab shows 9 items. The 'Body' section also includes tabs for 'Cookies', 'Schema', and 'Beautify'. At the bottom of the interface, the response details are shown: '201 Created' status, '27 ms' duration, '537 B' size, and a 'Save Response' button.

Body Cookies Headers (12) Test Results | ⚡

{ } JSON ▾ ▶ Preview ▾ Visualize ▾

```
1 {  
2   "id": 2,  
3   "content": "Post by regular user 2",  
4   "author_username": "regularuser2",  
5   "created_at": "2026-02-05T11:14:08.381411Z"  
6 }
```

201 Created ▾ 27 ms ▾ 537 B ▾ ⚡ Save Response ▾

11. Superuser Update Regular User Post (RBAC Fails - HTTPS)

<https://127.0.0.1:8000/api/posts/2/>

The screenshot shows a POST request in Postman. The URL is <https://127.0.0.1:8000/api/posts/2/>. The request method is PUT. The Body tab is selected, showing a raw JSON payload:

```
1 {  
2   "content": "Superuser trying to change regular user's post"  
3 }
```

The response status is 403 Forbidden. The response body is:

```
1 {  
2   "detail": "You do not have permission to perform this action."  
3 }
```

12. Regular User Update Own Post (RBAC Succeeds - HTTPS)

<https://127.0.0.1:8000/api/posts/>

The screenshot shows a POST request in the Postman application. The URL is `https://127.0.0.1:8000/api/posts/2/`. The request method is `PUT`. The body contains the following JSON:

```
1 {  
2   "content": "Regular user updates their own post"  
3 }
```

The response status is `200 OK`, with a response time of 23 ms and a size of 559 B. The response body is:

```
1 {  
2   "id": 2,  
3   "content": "Regular user updates their own post",  
4   "author_username": "regularuser2",  
5   "created_at": "2026-02-05T11:14:08.381411Z"  
6 }
```

13. Create Post (Unauthenticated Attempt - HTTPS) <https://127.0.0.1:8000/api/posts/>

The screenshot shows a POST request to <https://127.0.0.1:8000/api/posts/>. The request body contains a JSON object with a single key "content": "Attempt to post without a token.". The response status is 401 Unauthorized, with a response body indicating "Authentication credentials were not provided."

Request Body:

```
1 {  
2   "content": "Attempt to post without a token."  
3 }
```

Response Headers:

```
401 Unauthorized
```

Response Body:

```
1 {  
2   "detail": "Authentication credentials were not provided."  
3 }
```