

**TEST
CASES
AND
API TEST
FOR
MY BLOG APP**

Testing for User Registration

Test Case ID: TC-001					
Module Name: Registration					
Test Title: User Registration					
Description: Test for user registration functionality.					
Pre-condition: User does not exist in the system					
Post-condition: User account is created					
Test Steps:					
<ul style="list-style-type: none"> a. Navigate to "http://localhost:5173/register" b. Fill in registration details (name, email, password) c. Click on "Submit" button d. Complete registration 					
Test Case	Test Scenario	Test Data	Expected Results	Actual Results	Status
1	Valid registration	Name: Nishan Email:nishan@gmail.com Password: Password123	Registration form submits.	Same as expected	Pass
2	Email already exists	Name: Nishan Email:nishan@gmail.com Password: Password123	The form cannot be submitted.	Same as expected	Pass

Testing for User Login

Test Case ID: TC-002					
Module Name: Login					
Test Title: User Login					
Description: Test for login functionality					
Pre-condition: User already exists on the system					
Post-condition: User account is created					
Test Steps: a. Run the application by browsing to “ http://localhost:3000/login ” b. Enter login details and click on Login button					
Test case	Test Scenario	Test Data	Expected Results	Actual Results	Status
1	Valid credential	Email:nishan@gmail.com Password: Password123	Authentication success and homepage loads	Same as expected	Pass
2	Invalid/ Wrong password	Email:nishan@gmail.com Password: Password1234	Authentication failure and no action is performed.	Same as expected	Pass

API TESTING

This is performed using POSTMAN application locally in my device.

A) USER

1. Register a new user.

The screenshot shows the Postman interface for a 'register' endpoint. The request method is 'POST' to 'http://localhost:5000/api/v1/user/register'. The 'Body' tab is selected with 'raw' JSON content:

```
1 {
2     "username": "Nishan",
3     "email": "nishan@gmail.com",
4     "password": "Password123",
5     "blogs": []
6 }
```

The response status is '201 Created' with a response time of 236 ms and a body size of 603 B. The response JSON is:

```
1 {
2     "success": true,
3     "message": "New user created successfully",
4     "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY5MjlingM0ZTAyNWE4MTgwOWI1YWewZCIsImhdCI6MTc2NDM0MDkzMiw",
5     "user": {
6         "_id": "6929b4c4e025a81809b5aa0d",
7         "username": "Nishan",
8         "email": "nishan@gmail.com"
9     }
10 }
```

2. Login

The screenshot shows the Postman interface for a 'login' endpoint. The request method is 'POST' to 'http://localhost:5000/api/v1/user/login'. The 'Body' tab is selected with 'raw' JSON content:

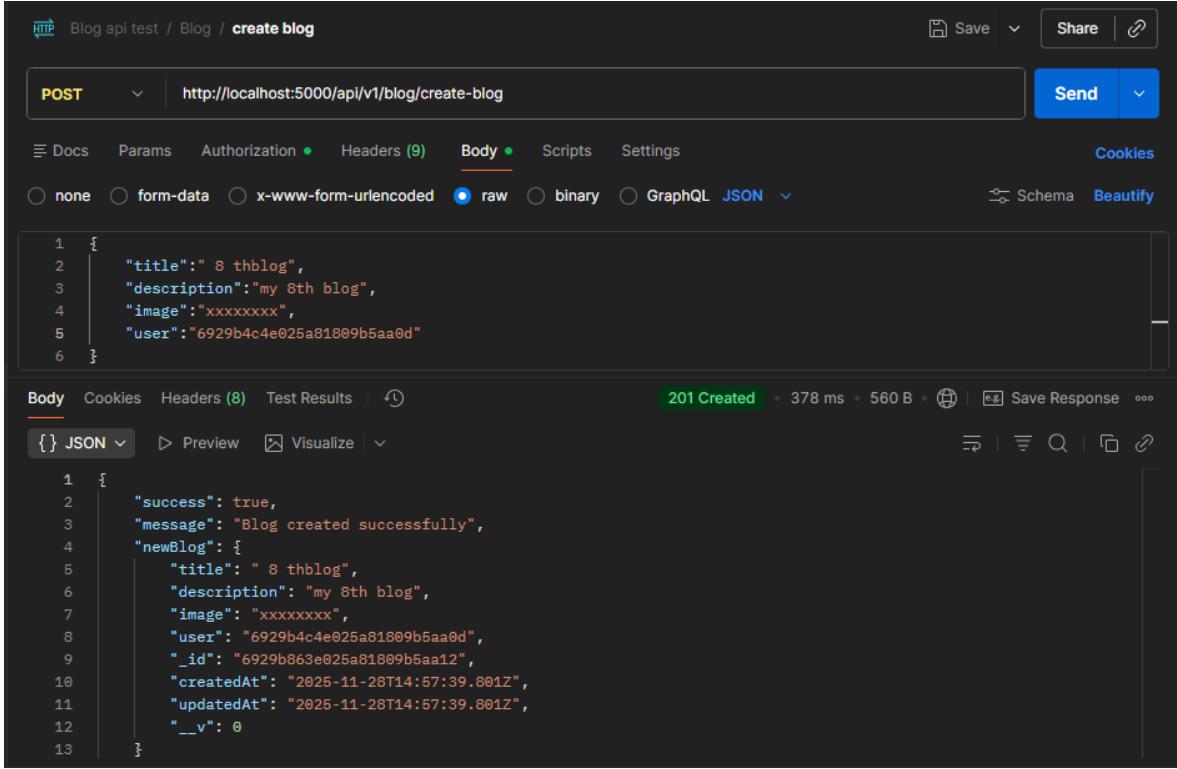
```
1 {
2     "email": "nishan@gmail.com",
3     "password": "Password123"
4 }
```

The response status is '200 OK' with a response time of 137 ms and a body size of 587 B. The response JSON is:

```
1 {
2     "success": true,
3     "message": "login successfully",
4     "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY5MjlingM0ZTAyNWE4MTgwOWI1YWewZCIsImhdCI6MTc2NDM0MTA0NSw",
5     "user": {
6         "_id": "6929b4c4e025a81809b5aa0d",
7         "username": "Nishan",
8         "email": "nishan@gmail.com"
9     }
10 }
```

B) BLOGS

1) Create a blog



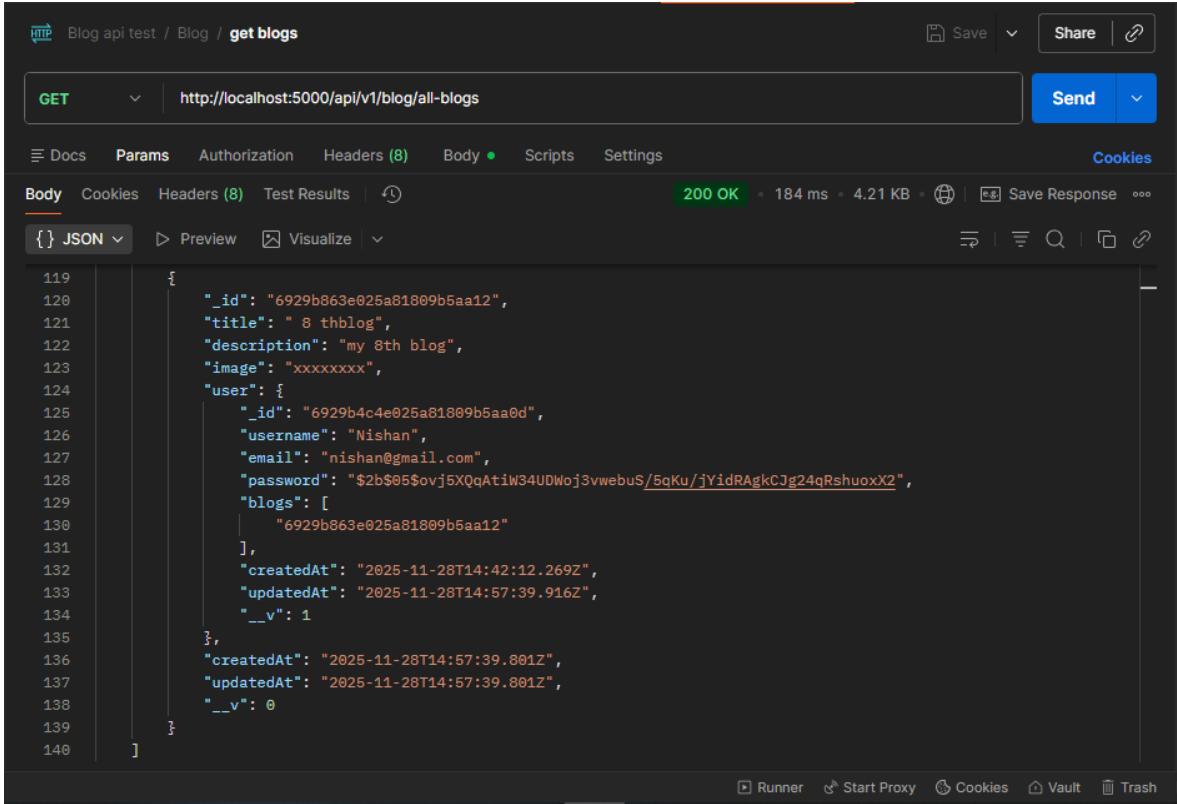
The screenshot shows the Postman interface for a POST request to `http://localhost:5000/api/v1/blog/create-blog`. The request body is raw JSON:

```
1 {  
2   "title": " 8 thblog",  
3   "description": "my 8th blog",  
4   "image": "xxxxxxxxx",  
5   "user": "6929b4c4e025a81809b5aa0d"  
6 }
```

The response status is 201 Created, with a response body containing the newly created blog details:

```
1 {  
2   "success": true,  
3   "message": "Blog created successfully",  
4   "newBlog": {  
5     "title": " 8 thblog",  
6     "description": "my 8th blog",  
7     "image": "xxxxxxxxx",  
8     "user": "6929b4c4e025a81809b5aa0d",  
9     "_id": "6929b863e025a81809b5aa12",  
10    "createdAt": "2025-11-28T14:57:39.801Z",  
11    "updatedAt": "2025-11-28T14:57:39.801Z",  
12    "__v": 0  
13  }
```

2) Get all blogs



The screenshot shows the Postman interface for a GET request to `http://localhost:5000/api/v1/blog/all-blogs`. The response status is 200 OK, with a response body containing a list of blogs, including one by user Nishan:

```
119 [ {  
120   "_id": "6929b863e025a81809b5aa12",  
121   "title": " 8 thblog",  
122   "description": "my 8th blog",  
123   "image": "xxxxxxxxx",  
124   "user": {  
125     "_id": "6929b4c4e025a81809b5aa0d",  
126     "username": "Nishan",  
127     "email": "nishan@gmail.com",  
128     "password": "$2b$05$ovj5XQqAtiW34UDWoj3vwebuS/5qKu/jYidRAgkCJg24qRshuoX2",  
129     "blogs": [  
130       "6929b863e025a81809b5aa12"  
131     ],  
132     "createdAt": "2025-11-28T14:42:12.269Z",  
133     "updatedAt": "2025-11-28T14:57:39.916Z",  
134     "__v": 1  
135   },  
136   "createdAt": "2025-11-28T14:57:39.801Z",  
137   "updatedAt": "2025-11-28T14:57:39.801Z",  
138   "__v": 0  
139 } ]
```

3) Update a Blog

The screenshot shows the Postman interface for a PUT request to update a blog. The URL is `http://localhost:5000/api/v1/blog/update-blog/6929b863e025a81809b5aa12`. The response status is 200 OK with a response time of 116 ms and a body size of 556 B. The response JSON is:

```
1 {  
2     "success": true,  
3     "message": "Blog updated successfully",  
4     "blog": {  
5         "_id": "6929b863e025a81809b5aa12",  
6         "title": "updated title",  
7         "description": "my 8th blog",  
8         "image": "xxxxxxxx",  
9         "user": "6929b4c4e025a81809b5aa0d",  
10        "createdAt": "2025-11-28T14:57:39.801Z",  
11        "updatedAt": "2025-11-28T15:01:11.838Z",  
12        "__v": 0  
13    }  
14 }
```

4) Delete a blog

The screenshot shows the Postman interface for a DELETE request to delete a blog. The URL is `http://localhost:5000/api/v1/blog/delete-blog/6929b863e025a81809b5aa12`. The response status is 200 OK with a response time of 315 ms and a body size of 321 B. The response JSON is:

```
1 {  
2     "success": true,  
3     "message": "Blog deleted successfully"  
4 }
```