

**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"><li>a. Navigate to "http://localhost:5173/register"</li><li>b. Fill in registration details (name, email, password)</li><li>c. Click on "Submit" button</li><li>d. Complete registration</li></ul>					
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

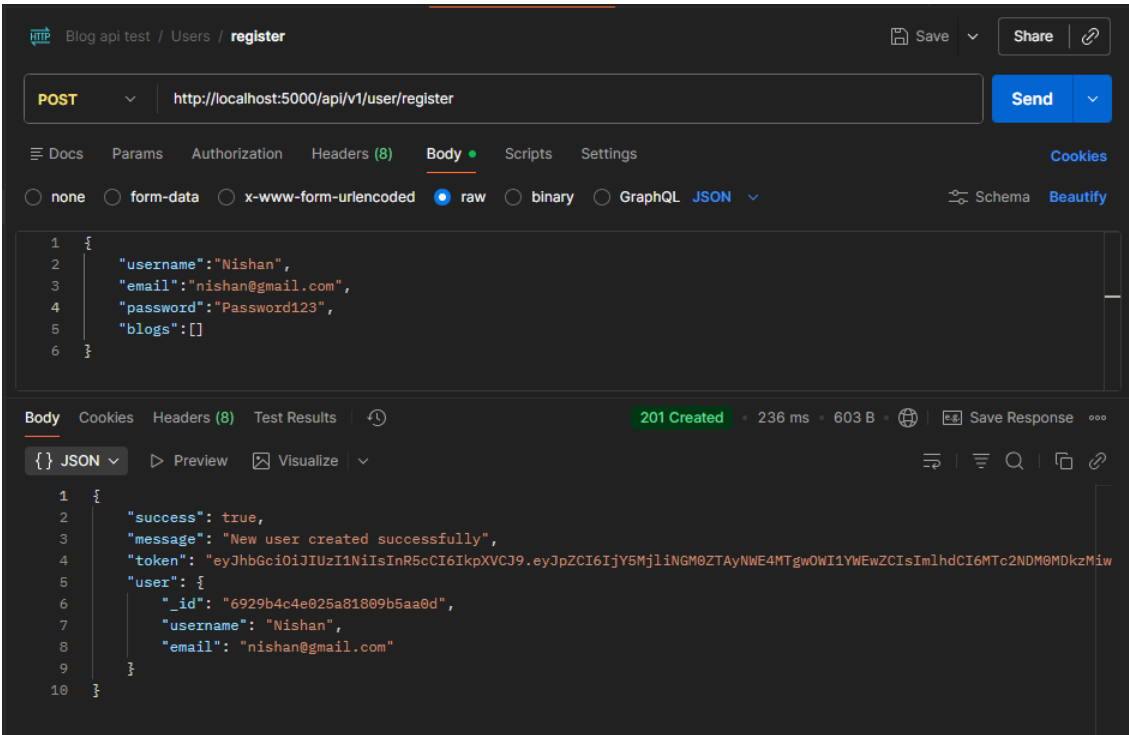
Test Case ID: TC-002					
Module Name: Login					
Test Title: User Login					
Description: Test for login functionality					
Pre-condition: User already exists on the system					
Post-condition: User account is created					
Test Steps: a. Run the application by browsing to “http://localhost:3000/login” 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

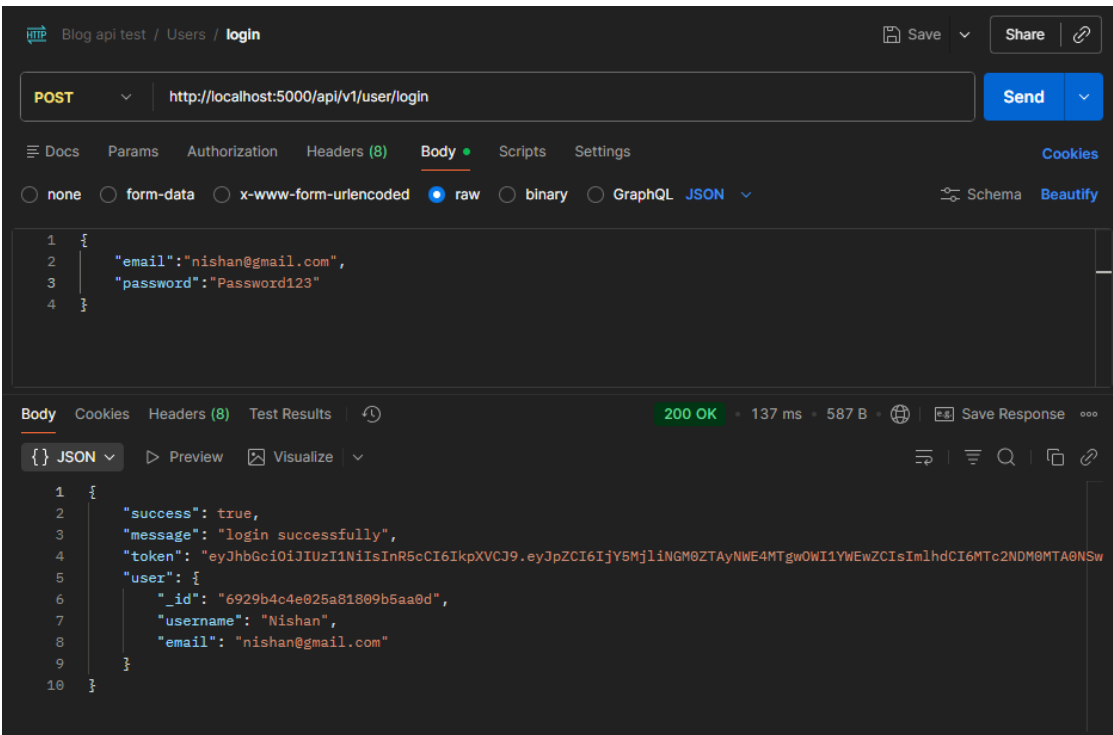
**This is performed using POSTMAN application locally in my device.**

### A) USER

### 1. Register a new user.



## 2. Login



## B) BLOGS

### 1) Create a blog

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

```
{
  "title": " 8 thblog",
  "description": "my 8th blog",
  "image": "xxxxxxx",
  "user": "6929b4c4e025a81809b5aa0d"
}
```

The response is a 201 Created status with a 378 ms response time and 560 B of data. The response body is JSON, showing a successful creation of a new blog entry with a new ID and timestamps.

```
{
  "success": true,
  "message": "Blog created successfully",
  "newBlog": {
    "title": " 8 thblog",
    "description": "my 8th blog",
    "image": "xxxxxxx",
    "user": "6929b4c4e025a81809b5aa0d",
    "_id": "6929b863e025a81809b5aa12",
    "createdAt": "2025-11-28T14:57:39.801Z",
    "updatedAt": "2025-11-28T14:57:39.801Z",
    "__v": 0
  }
}
```

### 2) Get all blogs

The screenshot shows a Postman interface for a REST client. The request is a GET to `http://localhost:5000/api/v1/blog/all-blogs`. The response is a 200 OK status with a 184 ms response time and 4.21 KB of data. The response body is JSON, showing a list of all blogs. The first blog entry is the one created in the previous step.

```
[
  {
    "_id": "6929b863e025a81809b5aa12",
    "title": " 8 thblog",
    "description": "my 8th blog",
    "image": "xxxxxxx",
    "user": {
      "_id": "6929b4c4e025a81809b5aa0d",
      "username": "Nishan",
      "email": "nishan@gmail.com",
      "password": "$2b$05$ovj5XQqAtiW34UDWoj3vwebuS/5qKu/jYidRAgkCJg24qRshuoxX2",
      "blogs": [
        "6929b863e025a81809b5aa12"
      ],
      "createdAt": "2025-11-28T14:42:12.269Z",
      "updatedAt": "2025-11-28T14:57:39.916Z",
      "__v": 1
    },
    "createdAt": "2025-11-28T14:57:39.801Z",
    "updatedAt": "2025-11-28T14:57:39.801Z",
    "__v": 0
  }
]
```

### 3) Update a Blog

The screenshot shows a REST client interface for a "Blog api test". The selected endpoint is "update blog" with a URL of `http://localhost:5000/api/v1/blog/update-blog/6929b863e025a81809b5aa12`. The method is set to **PUT**. The "Authorization" tab is active, showing "Bearer Token" as the auth type and `{{auth_token}}` as the token. The response status is **200 OK** with a response time of 116 ms and a body size of 556 B. The response body is displayed in JSON format:

```
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 a REST client interface for a "Blog api test". The selected endpoint is "delete blog" with a URL of `http://localhost:5000/api/v1/blog/delete-blog/6929b863e025a81809b5aa12`. The method is set to **DELETE**. The "Authorization" tab is active, showing "Bearer Token" as the auth type and `{{auth_token}}` as the token. The response status is **200 OK** with a response time of 315 ms and a body size of 321 B. The response body is displayed in JSON format:

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