

## API Testing Documentation

---

### Auth API

---

#### 1.1 Register User

- **Endpoint:** `http://localhost:8000/api/auth/register`
- **Method:** POST
- **Description:** Register a new user. Required fields: name, phone, email, password, confirmPassword.
- **Headers:** None
- **Body:**

```
{  
  "name": "Aniket Joshi",  
  "phone": "1234567890",  
  "email": "aniket@example.com",  
  "password": "Password123!",  
  "confirmPassword": "Password123!"  
}
```

- **Response Example (201 Created):**

```
{  
  "success": true,  
  "data": {  
    "user": {  
      "_id": "<userId>",  
      "name": "Aniket Joshi",  
      "phone": "1234567890",  
      "email": "aniket@example.com"  
    },  
    "tokens": {  
      "accessToken": "...",  
      "refreshToken": "..."  
    }  
  }  
}
```

```

    },
    "message": "User registered successfully"
}

```

- **Screenshot:**

The screenshot shows a POST request to `http://localhost:8000/api/auth/register`. The response status is `201 Created`, and the response body is a JSON object. The JSON object contains the following data:

```

1  {
2   "success": true,
3   "data": {
4     "user": {
5       "name": "Aniket Joshi",
6       "email": "aniket12@example.com",
7       "phone": "1234567999",
8       "password": "$2b$10$JvIHuQ9aKKjL0dpbWjQ00U.I72w2.6V26AV70xoyBHRexC3JPN/S",
9       "isSubscribed": false,
10      "otp": null,
11      "lastLoginTime": null,
12      "notificationPreferenceId": null,
13      "currentTheme": "light",
14      "isEmailVerified": false,
15      "address": "",
16      "marketExperience": "beginner",
17      "photo": "",
18      "_id": "6914b79bb62920cfa335e29d",
19      "createdAt": "2025-11-12T16:36:43.107Z",
20      "updatedAt": "2025-11-12T16:36:43.107Z",
21      "__v": 0
22    },
23    "tokens": {
24      "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJlc2VySWQ10Ii20TEBYj5Yjh1MjkYGNmNDMzNwUyOwQ1LC3pYXQ1OjE3NjI5NjU0MDMsImV4cCI6MTc2Mjk2NjMmM30.
kmhPleB-J7jZNqckET4X2mail1.Ss4R2mHxf7wJBka0",
25      "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9."
    }
  }

```

## 1.2 Login User

- **Endpoint:** `http://localhost:8000/api/auth/login`
- **Method:** POST
- **Description:** Login user with email and password to receive JWT tokens.
- **Headers:** None
- **Body:**

```
{
  "email": "aniket@example.com",
  "password": "Password123!"
}
```

- **Response Example (200 OK):**

```
{
  "success": true,
  "data": {
    "user": {

```

```

    "id": "<userId>",

    "name": "Aniket Joshi",

    "phone": "1234567890",

    "email": "aniket@example.com"

},

"tokens": {

    "accessToken": "...",

    "refreshToken": "..."

}

},

"message": "Login successful"
}

```

- **Screenshot:**

The screenshot shows a Postman request to `http://localhost:8000/api/auth/login` using the POST method. The response is a 200 OK status with a response time of 136 ms and a body size of 804 B. The response JSON is as follows:

```

1  {
2     "success": true,
3     "data": {
4         "user": {
5             "id": "6914aa2f22b07fdaa8d54e4bb",
6             "name": "Aniket Joshi",
7             "phone": "1234567999",
8             "email": "aniket11@example.com"
9         },
10        "tokens": {
11            "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJpc2VvSjQ1O1I2OTEwNWeyZjIyA3ZmRhYThKNTRlOGiILCJpYXQ1OjE3NjI5NjM4NjYsImV4cCI6MTc2Mjk2NDc2Nn0.
y4hOHViR0FnyHABb5zn1bSqAL1PtG4JqyOTAwB5mSY",
12            "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJpc2VvSjQ1O1I2OTEwNWeyZjIyYjA3ZmRhYThKNTRlOGiILCJpYXQ1OjE3NjI5NjM4NjYsImV4cCI6MTc2MuU20DY2Nn0.
yEc0-YICPu-RSM11q89PH4ppA1PJ_K4ucgsBhqxFM"
13        }
14    },
15    "message": "Login successful"
16 }

```

### 1.3 Forgot Password

- **Endpoint:** `http://localhost:8000/api/auth/forgot-password`
- **Method:** POST
- **Description:** Generate OTP for password reset.
- **Headers:** None
- **Body:**

```
{

```

```
        "email": "aniket@example.com"  
    }  

```

- **Response Example (200 OK):**

```
{  
    "success": true,  
    "data": {  
        "email": "aniket@example.com",  
        "otp": "123456"  
    },  
    "message": "OTP generated successfully"  
}  

```

- **Screenshot:**

The screenshot shows a POST request to `http://localhost:8000/api/auth/forgot-password`. The request body is a JSON object:

```
1 {  
2     "success": true,  
3     "data": {  
4         "email": "aniket@example.com",  
5         "otp": "123456"  
6     },  
7     "message": "OTP generated successfully"  
8 }
```

The response is a 200 OK status with the following JSON body:

```
1 {  
2     "success": true,  
3     "data": {  
4         "email": "aniket@example.com",  
5         "otp": "123456"  
6     },  
7     "message": "OTP generated successfully"  
8 }
```

## 1.4 Reset Password

- **Endpoint:** `http://localhost:8000/api/auth/reset-password`
- **Method:** POST
- **Description:** Reset password using OTP.
- **Headers:** None
- **Body:**

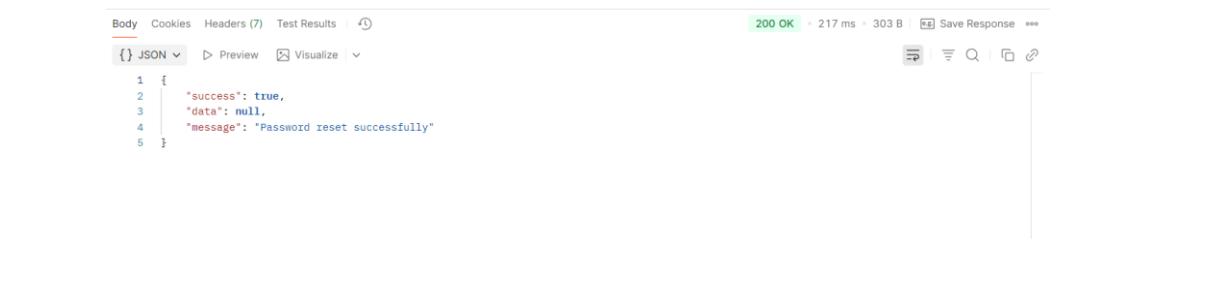
```
{  
    "email": "aniket@example.com",  
}
```

```
"otp": "123456",
"newPassword": "NewPass123!",
"confirmPassword": "NewPass123!"}
```

- **Response Example (200 OK):**

```
{  
  "success": true,  
  "data": null,  
  "message": "Password reset successfully"  
}
```

- **Screenshot:**



## 1.5 Change Password

- **Endpoint:** `http://localhost:8000/api/auth/change-password`
  - **Method:** POST
  - **Description:** Change password for logged-in user.
  - **Headers:**

Authorization: Bearer <accessToken>

- **Body:**

```
{  
  "oldPassword": "Password123!",  
  "newPassword": "NewPass123!",  
  "confirmPassword": "NewPass123!"  
}
```

- **Response Example (200 OK):**

```
{
  "success": true,
  "data": null,
  "message": "Password changed successfully"
}
```

- **Screenshot:**

The screenshot shows the Postman interface. At the top, it says "POST" and "http://localhost:8000/api/auth/change-password". Below that is a toolbar with "Params", "Authorization", "Headers (9)", "Body", "Scripts", and "Settings". On the right, there's a "Send" button and a "Cookies" section. Under "Body", there's a table for "Query Params" with columns "Key", "Value", "Description", and "Bulk Edit". The table has one row with "Key" and "Value" empty. At the bottom, the response tab is selected, showing a 200 OK status, 172 ms duration, 305 B size, and a "Save Response" button. The response body is displayed as JSON:

```
{}
  1  {
  2    "success": true,
  3    "data": null,
  4    "message": "Password changed successfully"
  5 }
```

## 1.6 Verify OTP

- **Endpoint:** `http://localhost:8000/api/auth/verify-otp`
- **Method:** POST
- **Description:** Verify OTP sent to email (temporary fixed OTP).
- **Headers:** None
- **Body:**

```
{
  "email": "aniket@example.com",
  "otp": "123456"
}
```

- **Response Example (200 OK):**

```
{
  "success": true,
  "data": {
```

```

    "email": "aniket@example.com",
    "verified": true
},
"message": "OTP verified successfully"
}

```

- **Screenshot:**

The screenshot shows a Postman interface. At the top, it says "POST" and "http://localhost:8000/api/otp/verify-otp". Below that is a table for "Query Params" with one row: "Key" (Value) and "Description". Under the "Body" tab, the JSON content is:

```

1 {
2   "success": true,
3   "data": {
4     "email": "aniket@example.com",
5     "verified": true
6   },
7   "message": "OTP verified successfully"
8 }

```

The response section shows a "200 OK" status with 15 ms response time and 344 B size. The response body is the same JSON object.

## OTP API

---

### 2.1 Register OTP (Request/Verify)

- **Endpoint:** <http://localhost:8000/api/otp/register>
- **Method:** POST
- **Description:**
  - If only **email** is provided → request OTP (temporary fixed OTP 123456).
  - If **email + otp** is provided → verify OTP and return JWT tokens.
  - If email does not exist → temporary user is created.
- **Headers:** None
- **Request Body (Request OTP):**

```
{
  "email": "aniket@example.com"
}
```

- **Response Example (200 OK - OTP Sent):**

```
{  
  "success": true,  
  "data": {  
    "email": "aniket@example.com",  
    "otp": "123456"  
  },  
  "message": "OTP sent successfully. (For testing OTP is 123456)"  
}
```

- **Request Body (Verify OTP):**

```
{  
  "email": "aniket@example.com",  
  "otp": "123456"  
}
```

- **Response Example (200 OK - OTP Verified):**

```
{  
  "success": true,  
  "data": {  
    "user": {  
      "id": "<userId>",  
      "email": "aniket@example.com"  
    },  
    "tokens": {  
      "accessToken": "...",  
      "refreshToken": "..."  
    }  
  },  
  "message": "OTP verified successfully"  
}
```

- **Screenshot:**

The screenshot shows three separate API requests in Postman:

- Request 1:** POST http://localhost:8000/api/otp/register. Response status: 200 OK. Response body (JSON):

```

1  {
2      "success": true,
3      "data": {
4          "email": "aniket@example.com",
5          "otp": "123456"
6      },
7      "message": "OTP sent successfully. (For testing OTP is 123456)"
8  }

```

- Request 2:** POST http://localhost:8000/api/otp/register. Response status: 200 OK. Response body (JSON):

```

1  {
2      "success": true,
3      "data": {
4          "user": {
5              "id": "6914aa2f22b07fd8a8d54e8b",
6              "email": "aniket@example.com"
7          },
8          "tokens": [
9              {
10                  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCIkVJC9.",
11                  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCIkVJC9."
12              }
13          ],
14          "message": "OTP verified successfully"
15      }
16  }

```

- Request 3:** POST http://localhost:8000/api/otp/login. Response status: 200 OK. Response body (JSON):

```

1  {
2      "success": true,
3      "data": {
4          "user": {
5              "id": "6914aa2f22b07fd8a8d54e8b",
6              "email": "aniket@example.com"
7          },
8          "tokens": [
9              {
10                  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCIkVJC9.",
11                  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCIkVJC9."
12              }
13          ],
14          "message": "OTP verified successfully"
15      }
16  }

```

## 2.2 Login OTP (Request/Verify)

- **Endpoint:** <http://localhost:8000/api/otp/login>
- **Method:** POST
- **Description:**
  - If only **email** is provided → request OTP.
  - If **email + otp** is provided → verify OTP and return JWT tokens.
  - User must already exist.
- **Headers:** None
- **Request Body (Request OTP):**

{

```
"email": "aniket@example.com"
}



- Response Example (200 OK - OTP Sent):


{
  "success": true,
  "data": {
    "email": "aniket@example.com",
    "otp": "123456"
  },
  "message": "Login OTP sent successfully. (For testing OTP is 123456)"
}



- Request Body (Verify OTP):


{
  "email": "aniket@example.com",
  "otp": "123456"
}



- Response Example (200 OK - OTP Verified):


{
  "success": true,
  "data": {
    "user": {
      "id": "<userId>",
      "email": "aniket@example.com"
    },
    "tokens": {
      "accessToken": "...",
      "refreshToken": "..."
    }
  },
  "message": "Login OTP verified successfully"
}
```

- **Screenshot:**

The screenshot shows two API requests in Postman:

**Request 1: POST http://localhost:8000/api/otp/login**

Body (JSON):

```
{
  "success": true,
  "data": {
    "email": "aniket@example.com",
    "otp": "123456"
  },
  "message": "Login OTP sent successfully. (For testing OTP is 123456)"
}
```

**Response 1: 200 OK**

**Request 2: POST http://localhost:8000/api/otp/login**

Body (JSON):

```
{
  "success": true,
  "data": {
    "user": {
      "id": "6914aa2f22b07fd8d54e8b",
      "email": "aniket@example.com"
    },
    "tokens": [
      "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiI2OTk4YWEyZjIyYjA3MWhhTmNTRl0GiiLCJpYXQiOjE3NjI5NjQyMTksImV4cCI6MTc2MjK2NTEx0X0.eyJ1c2VySWQiOiI2OTk4YWEyZjIyYjA3MWhhTmNTRl0GiiLCJpYXQiOjE3NjI5NjQyMTksImV4cCI6MTc2MjK2NTEx0X0.HJIMHh0t14AyxvssJHgkbe3WwA_C1jzkd8Pi32pM",
      "refreshToken": "eyJhbGciOiI1NU1hIisInR5cCI6IkpXVCJ9.eyJpc3MiOiI2OTk4YWEyZjIyYjA3MWhhTmNTRl0GiiLCJpYXQiOjE3NjI5NjQyMTksImV4cCI6MTc2MzU20TAx0X0.RFNL-jT1Vp51Bp8SiNv1QdtCNzneMqjW51bq10DS9K"
    ]
  },
  "message": "Login OTP verified successfully"
}
```

**Response 2: 200 OK**

## 2.3 Verify OTP Directly

- **Endpoint:** <http://localhost:8000/api/otp/verify-otp>
- **Method:** POST
- **Description:** Verify OTP directly using email + OTP.
- **Headers:** None
- **Body:**

```
{
  "email": "aniket@example.com",
  "otp": "123456"
```

}

- **Response Example (200 OK):**

```
{  
  "success": true,  
  "data": {  
    "email": "aniket@example.com",  
    "verified": true  
  },  
  "message": "OTP verified successfully"  
}
```

- **Screenshot:**

The screenshot shows the Postman interface. At the top, there's a header bar with 'POST' selected, the URL 'http://localhost:8000/api/otp/verify-otp', and a 'Send' button. Below the header are tabs for 'Params', 'Authorization', 'Headers (8)', 'Body', 'Scripts', and 'Settings'. The 'Params' tab is active. Under 'Query Params', there's a table with one row containing 'Key' and 'Value' columns. The main body of the screenshot shows a successful response from the API. The status bar at the bottom indicates '200 OK' with a green background, '11 ms', '344 B', and a 'Save Response' button.

```
200 OK 11 ms 344 B Save Response  
{} JSON ▾ Preview Visualize ▾  
1 {  
2   "success": true,  
3   "data": {  
4     "email": "aniket@example.com",  
5     "verified": true  
6   },  
7   "message": "OTP verified successfully"  
8 }
```

## User API

### 3.1 Update User Profile

- **Endpoint:** `http://localhost:8000/api/user/update/<userId>`
- **Method:** POST
- **Description:** Update an existing user's profile. You can update **name**, **email**, or **phone**. At least one field should be provided.
- **Headers:**

Authorization: Bearer <accessToken>

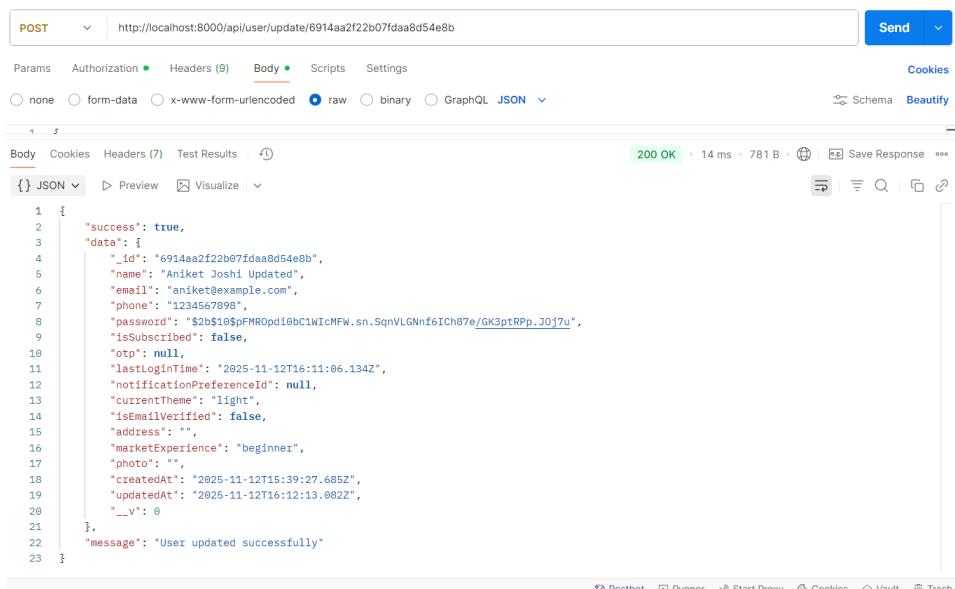
- **Request Body:**

```
{  
  "name": "Aniket Joshi Updated",  
  "email": "aniketupdated@example.com",  
  "phone": "9876543210"  
}
```

- **Response Example (200 OK):**

```
{  
  "success": true,  
  "data": {  
    "_id": "<userId>",  
    "name": "Aniket Joshi Updated",  
    "email": "aniketupdated@example.com",  
    "phone": "9876543210"  
  },  
  "message": "User updated successfully"  
}
```

- **Screenshot:**



### 3.2 Delete User Profile

- **Endpoint:** `http://localhost:8000/api/user/delete/<userId>`
- **Method:** `DELETE`
- **Description:** Delete a user profile by ID. Only logged-in users can perform this action.
- **Headers:**

Authorization: Bearer <accessToken>

- **Request Body:** None
- **Response Example (200 OK):**

```
{  
  "success": true,  
  "data": null,  
  "message": "User deleted successfully"  
}
```

- **Screenshot:**

The screenshot shows the Postman interface with a DELETE request to `http://localhost:8000/api/user/delete/6914a92222b07fdaa8d54e85`. The 'Headers' tab contains `Content-Type: application/json`. The 'Body' tab is empty. The 'Params' tab shows a single parameter `Key` with value `Value`.

The screenshot shows the Postman interface after sending the DELETE request. The status is `200 OK` with a response time of 24 ms and a size of 301 B. The response body is a JSON object:

```
{  
  "success": true,  
  "data": null,  
  "message": "User deleted successfully"  
}
```

## Notifications API

---

### 4.1 Create Notification

- **Endpoint:** `http://localhost:8000/api/notifications`
- **Method:** POST
- **Description:** Create a new notification. At least one of **photo, hyperlink, or description** is required.
- **Headers:**

Authorization: Bearer <accessToken>

- **Request Body:**

```
{  
  "photo": "https://example.com/image.jpg",  
  "hyperlink": "https://example.com",  
  "description": "This is a test notification"  
}
```

- **Response Example (201 Created):**

```
{  
  "success": true,  
  "data": {  
    "_id": "<notificationId>",  
    "photo": "https://example.com/image.jpg",  
    "hyperlink": "https://example.com",  
    "description": "This is a test notification",  
    "createdAt": "2025-11-12T10:00:00.000Z",  
    "updatedAt": "2025-11-12T10:00:00.000Z"  
  },  
  "message": "Notification created successfully"
```

}

- **Screenshot:**

The screenshot shows a POST request to the endpoint `http://localhost:8000/api/notifications`. The response status is `201 Created`, and the message is `"Notification created successfully"`.

Key	Value	Description
Key	Value	Description

```
1 {
2   "success": true,
3   "data": [
4     {
5       "_id": "6914b4718e2920cf4335e28d",
6       "photo": "https://example.com/image.jpg",
7       "hyperlink": "https://example.com",
8       "description": "This is a test notification",
9       "_id": "6914b4718e2920cf4335e28e",
10      "createdAt": "2025-11-12T16:23:13.931Z",
11      "updatedAt": "2025-11-12T16:23:13.931Z",
12      "_v": 0
13    },
14  ],
15  "message": "Notification created successfully"
16 }
```

## 4.2 Get All Notifications

- **Endpoint:** `http://localhost:8000/api/notifications`
- **Method:** GET
- **Description:** Fetch all notifications sorted by latest first.
- **Headers:** Optional (if your app requires auth)
- **Request Body:** None
- **Response Example (200 OK):**

```
{
  "success": true,
  "data": [
    {
      "_id": "<notificationId>",
      "photo": "https://example.com/image.jpg",
      "hyperlink": "https://example.com",
      "description": "This is a test notification",
      "createdAt": "2025-11-12T10:00:00.000Z",
      "updatedAt": "2025-11-12T10:00:00.000Z"
    }
  ]
}
```

```

    },
],
"message": "Notifications fetched successfully"
}

```

- **Screenshot:**

The screenshot shows a Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:8000/api/notifications
- Headers:** (6)
- Body:** ({} JSON) - Preview and Visualize tabs are visible.
- Response Status:** 200 OK
- Response Time:** 93 ms
- Response Size:** 1.25 KB
- Response Content:** (JSON data shown below)

```

1 {
2   "success": true,
3   "data": [
4     {
5       "_id": "6914b4710e2920cf4335e28e",
6       "id": "6914b4710e2920cf4335e28d",
7       "photo": "https://example.com/image.jpg",
8       "hyperlink": "https://example.com",
9       "description": "This is a test notification",
10      "createdAt": "2025-11-12T16:23:13.931Z",
11      "updatedAt": "2025-11-12T16:23:13.931Z",
12      "__v": 0
13    },
14    {
15      "_id": "69149bcbf38b97860eb5c940",
16      "id": "69149bcbf38b97860eb5c93f",
17      "photo": "https://example.com/image.jpg",
18      "hyperlink": "",
19      "description": "",
20      "createdAt": "2025-11-12T14:38:03.963Z",
21      "updatedAt": "2025-11-12T14:38:03.963Z",
22      "__v": 0
23    }
  ],
  "message": "Notifications fetched successfully"
}

```

### 4.3 Update Notification

- **Endpoint:** http://localhost:8000/api/notifications/update/<notificationId>
- **Method:** POST
- **Description:** Update a notification. At least one of **photo**, **hyperlink**, or **description** must be provided.
- **Headers:**

Authorization: Bearer <accessToken>

- **Request Body:**

```
{
  "description": "Updated notification description"
}
```

- **Response Example (200 OK):**

```
{
  "success": true,
```

```

"data": {

  "_id": "<notificationId>",

  "photo": "https://example.com/image.jpg",

  "hyperlink": "https://example.com",

  "description": "Updated notification description",

  "createdAt": "2025-11-12T10:00:00.000Z",

  "updatedAt": "2025-11-12T12:00:00.000Z"

},

"message": "Notification updated successfully"
}

```

- **Screenshot:**

The screenshot shows a POST request to the URL `http://localhost:8000/api/notifications/update/6914b4718e2920cf4335e28e`. The response is successful (200 OK) with a response time of 24 ms and a size of 582 B. The response body is a JSON object:

```

{
  "success": true,
  "data": {
    "_id": "6914b4718e2920cf4335e28e",
    "id": "6914b4718e2920cf4335e28d",
    "photo": "https://example.com/image.jpg",
    "hyperlink": "https://example.com",
    "description": "Updated notification description",
    "createdAt": "2025-11-12T16:23:13.931Z",
    "updatedAt": "2025-11-12T16:26:57.391Z",
    "__v": 0
  },
  "message": "Notification updated successfully"
}

```

#### 4.4 Delete Notification

- **Endpoint:** `http://localhost:8000/api/notifications/<notificationId>`
- **Method:** DELETE
- **Description:** Delete a notification by ID.
- **Headers:**

Authorization: Bearer <accessToken>

- **Request Body:** None
- **Response Example (200 OK):**

```
{

```

```
"success": true,  
"data": null,  
"message": "Notification deleted successfully"  
}
```

- **Screenshot:**

The screenshot shows a REST API testing interface with the following details:

- Method:** DELETE
- URL:** http://localhost:8000/api/notifications/6914b4718e2920cf4335e28e
- Status:** 200 OK
- Time:** 23 ms
- Size:** 309 B
- Buttons:** Send, Cookies, Save Response, Bulk Edit, Copy, Paste, Find, Refresh, Close.
- Params Tab:** Active, showing Headers (8), Body, Scripts, Settings.
- Query Params:** Key, Value, Description columns.
- Body Tab:** Active, showing Body, Cookies, Headers (7), Test Results, and a JSON preview.
- JSON Preview:** Shows the response body as JSON:

```
1 {  
2   "success": true,  
3   "data": null,  
4   "message": "Notification deleted successfully"  
5 }
```