Using Postman to Test CRUD Operations

This document provides sample requests to test all the routes in the Express application using Postman. The application allows creating, reading, updating, and deleting user data.

# 1. Create a New User

Method: POST

URL: http://192.168.0.107:3000/api/users

Body: JSON

{  
 "name": "John Doe",  
 "email": "john.doe@example.com"  
 }

Response:  
{

"message": "Successfully signed up!"

}

# 2. Read All Users

Method: GET

URL: [http://192.168.0.107:3000/api/users](http://localhost:3000/api/users)

RESPONSE:  
[

{

"\_id": 101,

"name": "John Doe",

"email": "john.doe@example.com",

"created": "2024-07-05T11:29:06.321Z",

"\_\_v": 0

}

]

# 3. Read a Single User by ID

Method: GET

URL: http://192.168.0.107:3000/api/users/USER\_ID

Replace USER\_ID with the actual user ID.

RESPONSE:

{

"\_id": 101,

"name": "John Doe",

"email": "john.doe@example.com",

"created": "2024-07-05T11:29:06.321Z",

"\_\_v": 0

}

# 4. Update a User

Method: PUT

URL: http://192.168.0.107:3000/api/users/USER\_ID

Body: JSON

{  
 "name": "Jane Doe",  
 "email": "jane.doe@example.com"  
 }

Replace USER\_ID with the actual user ID.

RESPONSE:  
  
{

"\_id": 101,

"name": "Jane Doe",

"email": "jane.doe@example.com",

"created": "2024-07-05T11:29:06.321Z",

"\_\_v": 0,

"updated": "2024-07-05T11:30:43.603Z"

}

# 5. Delete a User

Method: DELETE

URL: http://192.168.0.107:3000/api/users/USER\_ID

Replace USER\_ID with the actual user ID.

RESPONSE:  
  
{

"\_id": 101,

"name": "Jane Doe",

"email": "jane.doe@example.com",

"created": "2024-07-05T11:29:06.321Z",

"\_\_v": 0,

"updated": "2024-07-05T11:30:43.603Z"

}