

Echo Tree - Chat & Message Service + Note & Tree Hole Service API Actual Test Report

Tester: API Testing Suite

Base URL: http://localhost:3000

Services Tested: Chat & Message Service, Note & Tree Hole Service

Test User: admin (admin@echotree.com)

Token:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiI2OTFjZDIkZmJhNjg1NDM4ZTQzMDFiYjkiLCJpYXQiOiJlE3NjM1NDMzMjcslmV4cCI6MTc2NDE0ODEyN30.W9IvyMDbE5klkOYWam1XK6U9IRDvCANjINt8aWly30
user_id": "691cd9dfba685438e4301bb9"

Chat & Message Service API Tests

1. Create New Chat

Use Case

User creates a new private chat session with the tree

API Details

Resource: chats

HTTP Method: POST

Request URL: http://localhost:3000/api/chats

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

Request Body:

{

```
"title": "Testing Chat Service"

}
```

Expected Response:

- **Status Code:** 201 Created
- **Success:** true
- **Response Body:**

```
{

  "success": true,

  "data": {

    "user_id": "691cd9dfba685438e4301bb9",

    "start_time": "2025-11-19T09:09:53.331Z",

    "closed_at": null,

    "message_count": 0,

    "deleted_at": null,

    "_id": "691d8961ba685438e4301c62",

    "createdAt": "2025-11-19T09:09:53.332Z",

    "updatedAt": "2025-11-19T09:09:53.332Z",

    "__v": 0,

    "chat_id": "691d8961ba685438e4301c62",

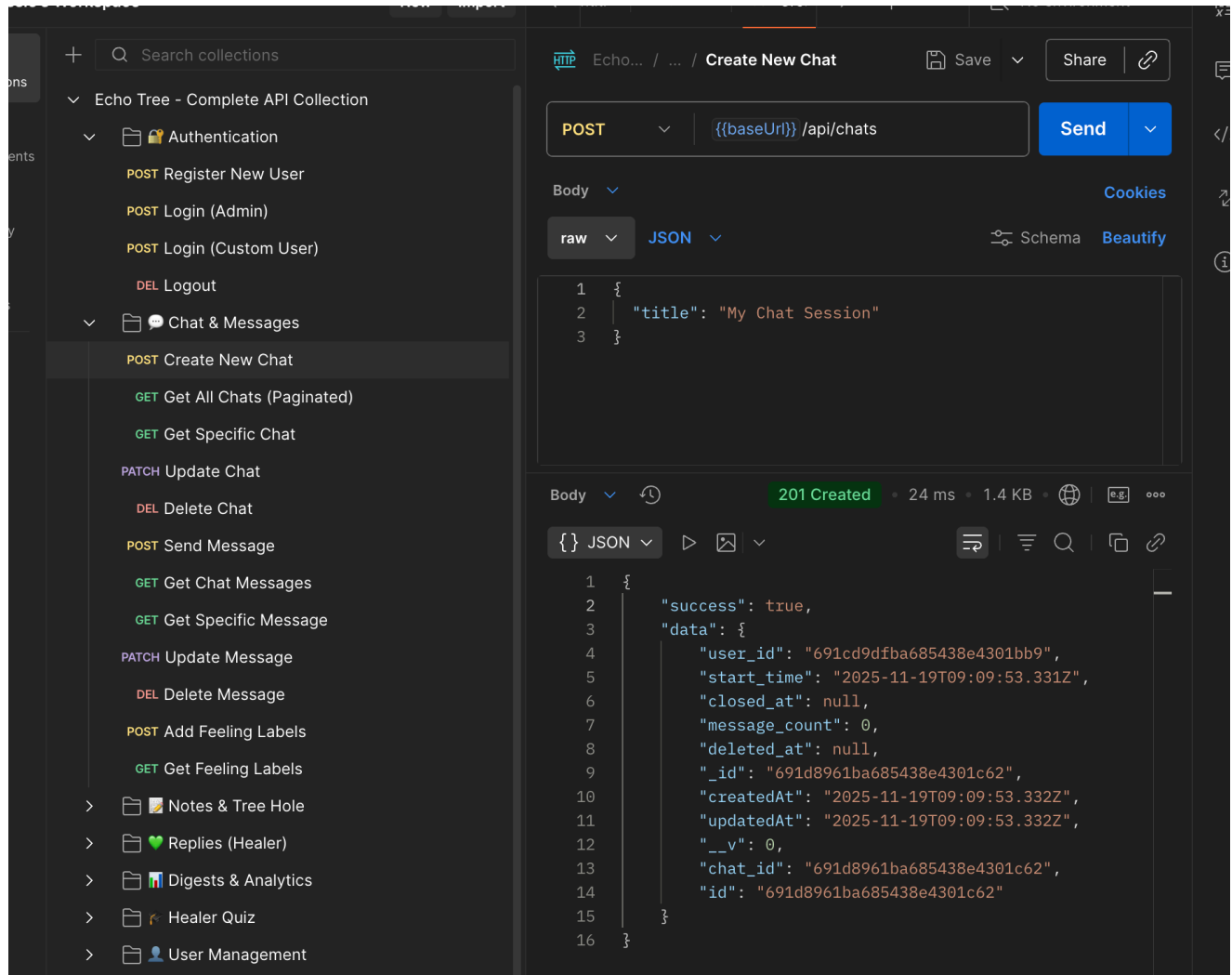
    "id": "691d8961ba685438e4301c62"

  }

}
```

Test Result:  PASS

Notes: Chat created successfully.



2. Send Message to Chat

Use Case

User sends a message to the tree in an existing chat session

API Details

Resource: messages (sub-resource of chats)

HTTP Method: POST

Request URL:

`http://localhost:3000/api/chats/691d0c77ba685438e4301c2d/messages`

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

Request Body:

```
{  
  
  "content": "Hello, wise tree. I need some guidance.",  
  
  "mood": 3,  
  
  "energy": 3  
  
}
```

Expected Response:

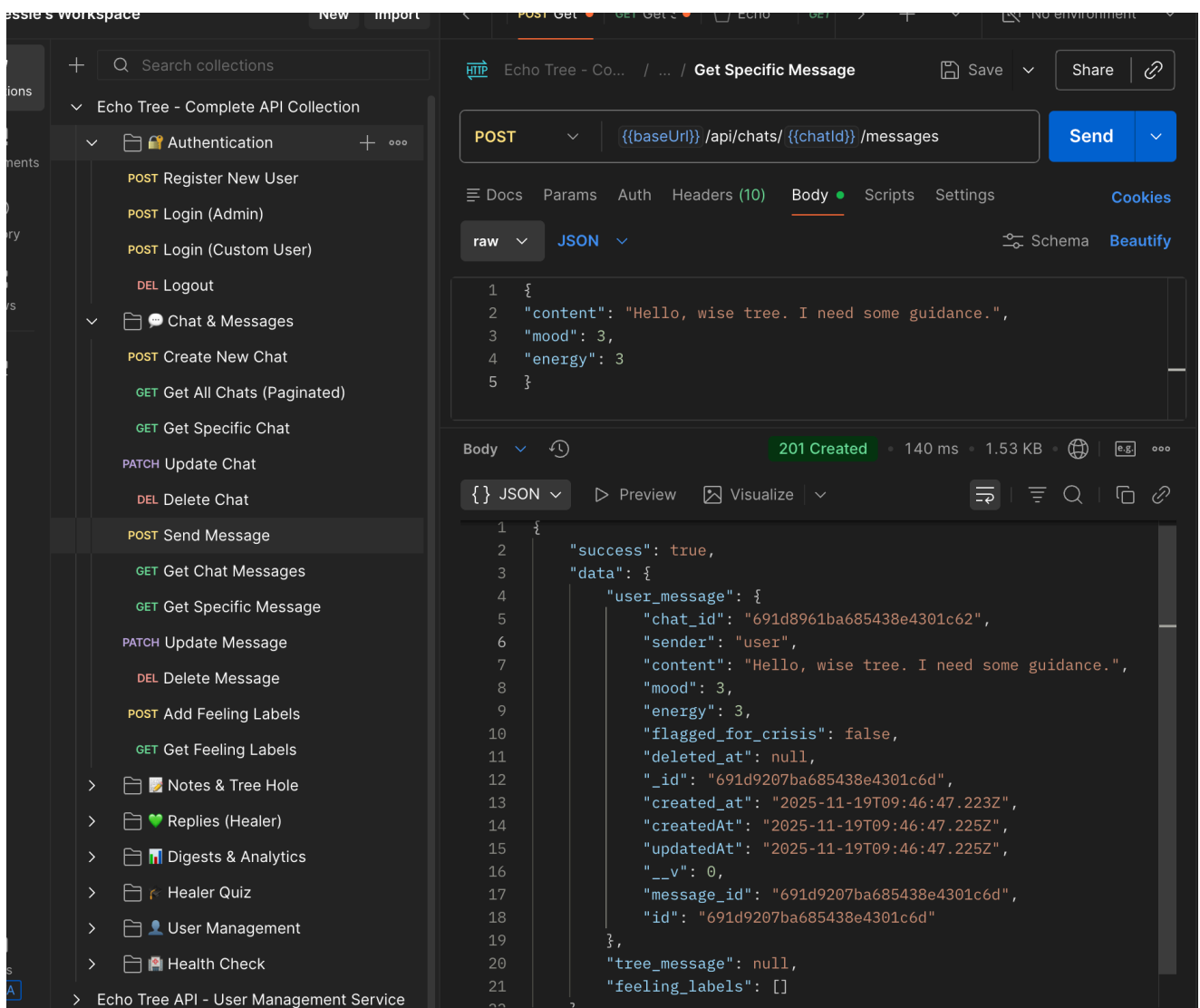
- **Status Code:** 201 Created
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": {  
  
    "user_message": {  
  
      "chat_id": "691d8961ba685438e4301c62",  
  
      "sender": "user",  
  
      "content": "Hello, wise tree. I need some guidance.",  
  
      "mood": 3,  
  
      "energy": 3,  
  
      "flagged_for_crisis": false,  
  
      "deleted_at": null,  
  
      "_id": "691d9207ba685438e4301c6d",  
  
    }  
  
  }  
  
}
```

```
"created_at": "2025-11-19T09:46:47.223Z",  
"createdAt": "2025-11-19T09:46:47.225Z",  
"updatedAt": "2025-11-19T09:46:47.225Z",  
"__v": 0,  
"message_id": "691d9207ba685438e4301c6d",  
"id": "691d9207ba685438e4301c6d"  
},  
"tree_message": null,  
"feeling_labels": []  
}  
}
```

Test Result:  **PASS**

Notes:



3. Get Chat Messages

Use Case

Retrieve all messages from a specific chat with pagination

API Details

Resource: messages (sub-resource of chats)

HTTP Method: GET

Request URL:

`http://localhost:3000/api/chats/691d0c77ba685438e4301c2d/messages?limit=10&offset=0`

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Query Parameters:

- `limit` : 10 (max results per page)
- `offset` : 0 (starting position)

Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": [  
  
    {  
  
      "_id": "691d9207ba685438e4301c6d",  
  
      "chat_id": "691d8961ba685438e4301c62",  
  
      "sender": "user",  
  
      "content": "Hello, wise tree. I need some guidance.",  
  
      "mood": 3,  
  
      "energy": 3,  
  
      "flagged_for_crisis": false,  
  
      "deleted_at": null,  
  
      "created_at": "2025-11-19T09:46:47.223Z",  
  
      "createdAt": "2025-11-19T09:46:47.225Z",  
  
      "updatedAt": "2025-11-19T09:46:47.225Z",  
  
      "__v": 0  
    }  
  ]  
}
```

```

],

"pagination": {

"total": 1,

"limit": 10,

"offset": 0

}

}

```

Test Result: ✅ **PASS**

Notes: Successfully retrieved messages with pagination metadata

Looks like your team is full. To expand, organize, manage your team effortlessly, [upgrade your plan](#).

essie's Workspace New Import < Get GET Get S Echo GET Get C > + ▼ No environment ▼

+ Q Search collections

▼ Echo Tree - Complete API Collection

▼ 📁 Authentication

POST Register New User

POST Login (Admin)

POST Login (Custom User)

DEL Logout

▼ 📁 Chat & Messages

POST Create New Chat

GET Get All Chats (Paginated)

GET Get Specific Chat

PATCH Update Chat

DEL Delete Chat

POST Send Message

GET Get Chat Messages

GET Get Specific Message

PATCH Update Message

DEL Delete Message

POST Add Feeling Labels

GET Get Feeling Labels

> 📁 Notes & Tree Hole

> 📁 ❤️ Replies (Healer)

> 📁 Digests & Analytics

> 📁 Healer Quiz

> 📁 User Management

> 📁 Health Check

HTTP Echo Tree - Comp... / ... / Get Chat Messages Save Share 🔗

GET ▼ {{baseUrl}} /api/chats/ {{chatId}} /messages?limit=10&... Send ▼

≡ Docs ● Params Auth Headers (8) Body Scripts Settings Cookies

Headers 👁 7 hidden

	Key	Value	...	Bulk Edit	Presets	▼
<input checked="" type="checkbox"/>	Authorization	Bearer {{token}}				
	Key	Value		Description		

Body ▼ 🔄 200 OK • 182 ms • 1.45 KB • 🌐 🔍 🔗 🔗

{} JSON ▶ Preview 🖼 Visualize ▼ 🔄 🔍 🔗 🔗

```

1 {
2   "success": true,
3   "data": [
4     {
5       "id": "691d9207ba685438e4301c6d",
6       "chat_id": "691d8961ba685438e4301c62",
7       "sender": "user",
8       "content": "Hello, wise tree. I need some guidance.",
9       "mood": 3,
10      "energy": 3,
11      "flagged_for_crisis": false,
12      "deleted_at": null,
13      "created_at": "2025-11-19T09:46:47.223Z",
14      "createdAt": "2025-11-19T09:46:47.225Z",
15      "updatedAt": "2025-11-19T09:46:47.225Z",
16      "__v": 0
17    }
18  ],

```

4. Add Feeling Labels to Message

Use Case

User adds emotional labels to their message for better tracking

API Details

Resource: feelings (sub-resource of messages)

HTTP Method: POST

Request URL:

```
http://localhost:3000/api/chats/691d0c77ba685438e4301c2d/messages/691d0cb2ba685438e4301c34/feelings
```

Request Headers:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

```
Content-Type: application/json
```

Request Body:

```
{  
  
  "labels": ["anxious", "hopeful", "grateful"]  
  
}
```

Expected Response:

- **Status Code:** 201 Created
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": [  
  
    {  
  
      "message_id": "691d9207ba685438e4301c6d",  

```

```
"label_text": "anxious",

"deleted_at": null,

"_id": "691ea193ba685438e4301c7b",

"created_at": "2025-11-20T05:05:23.290Z",

"createdAt": "2025-11-20T05:05:23.296Z",

"updatedAt": "2025-11-20T05:05:23.296Z",

"__v": 0,

"label_id": "691ea193ba685438e4301c7b",

"id": "691ea193ba685438e4301c7b"

},

{

"message_id": "691d9207ba685438e4301c6d",

"label_text": "hopeful",

"deleted_at": null,

"_id": "691ea193ba685438e4301c7d",

"created_at": "2025-11-20T05:05:23.330Z",

"createdAt": "2025-11-20T05:05:23.330Z",

"updatedAt": "2025-11-20T05:05:23.330Z",

"__v": 0,

"label_id": "691ea193ba685438e4301c7d",

"id": "691ea193ba685438e4301c7d"

},

{

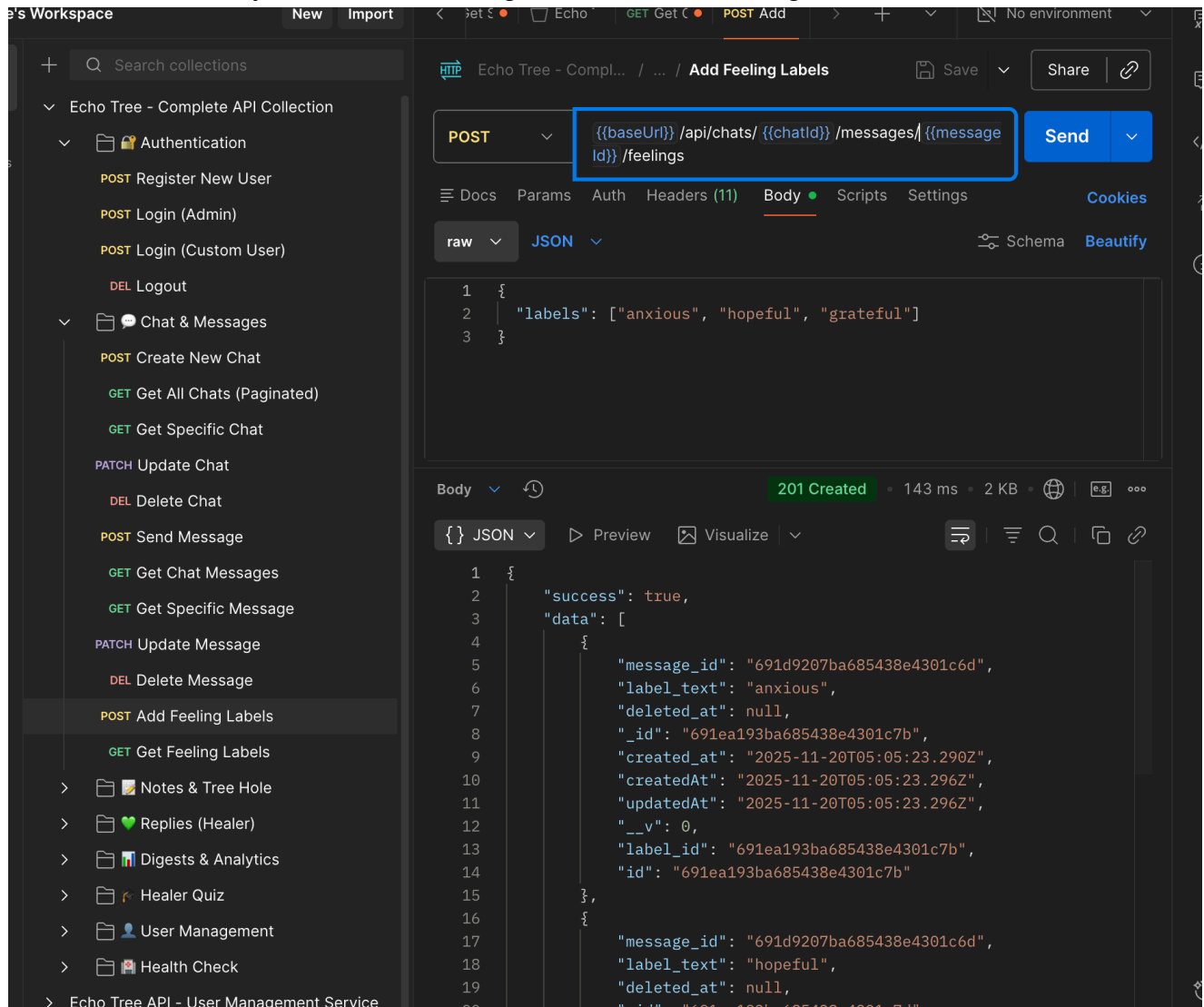
"message_id": "691d9207ba685438e4301c6d",

"label_text": "grateful",
```

```
"deleted_at": null,  
  
"_id": "691ea193ba685438e4301c7f",  
  
"created_at": "2025-11-20T05:05:23.333Z",  
  
"createdAt": "2025-11-20T05:05:23.333Z",  
  
"updatedAt": "2025-11-20T05:05:23.333Z",  
  
"__v": 0,  
  
"label_id": "691ea193ba685438e4301c7f",  
  
"id": "691ea193ba685438e4301c7f"  
  
}  
  
]  
  
}
```

Test Result:  PASS

Notes: Successfully created two feeling labels for the message



5. Get All Chats (with Pagination)

Use Case

Retrieve all chat sessions for the authenticated user with sorting and pagination

API Details

Resource: chats

HTTP Method: GET

Request URL: {{baseUrl}}/api/chats?

limit=10&offset=0&sortBy=updated_at&sortOrder=desc

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Query Parameters:

- `limit` : 10 (max results per page)
- `offset` : 0 (starting position)
- `sortBy` : `updated_at` (sort field)
- `sortOrder` : `desc` (descending order)

Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body:**

```
{  
  "success": true,  
  "data": [  
    {  
      "_id": "691d8961ba685438e4301c62",  
      "user_id": "691cd9dfba685438e4301bb9",  
      "start_time": "2025-11-19T09:09:53.331Z",  
      "closed_at": null,  
      "message_count": 1,  
      "deleted_at": null,  
      "createdAt": "2025-11-19T09:09:53.332Z",  
      "updatedAt": "2025-11-19T09:46:47.237Z",  
      "__v": 0  
    },  
    {  
      "_id": "691d0c77ba685438e4301c2d",
```

```
"user_id": "691cd9dfba685438e4301bb9",

"start_time": "2025-11-19T00:16:55.669Z",

"closed_at": null,

"message_count": 1,

"deleted_at": null,

"createdAt": "2025-11-19T00:16:55.669Z",

"updatedAt": "2025-11-19T00:17:55.128Z",

"__v": 0

}

],

"pagination": {

"total": 2,

"limit": 10,

"offset": 0

}

}
```

Test Result:  **PASS**

Notes:

The screenshot displays the Echo Tree API client interface. On the left, a sidebar shows a collection tree with categories like Authentication, Chat & Messages, Notes & Tree Hole, Replies (Healer), Digests & Analytics, Healer Quiz, User Management, and Health Check. The 'Get All Chats (Paginated)' endpoint is selected. The main panel shows the request details: a GET request to `{{baseUrl}}/api/chats?limit=10&offset=0&sortBy=updated_at`. The response is a 200 OK status with a 37 ms response time and 1.62 KB of data. The response body is a JSON array of chat objects, each containing fields like `_id`, `user_id`, `start_time`, `closed_at`, `message_count`, `deleted_at`, `createdAt`, `updatedAt`, and `__v`.

6. Get Feeling Labels for Message

Use Case

Retrieve all feeling labels associated with a specific message

API Details

Resource: feelings (sub-resource of messages)

HTTP Method: GET

Request URL:

`{{baseUrl}}/api/chats/{{chatId}}/messages/{{messageId}}/feelings`

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

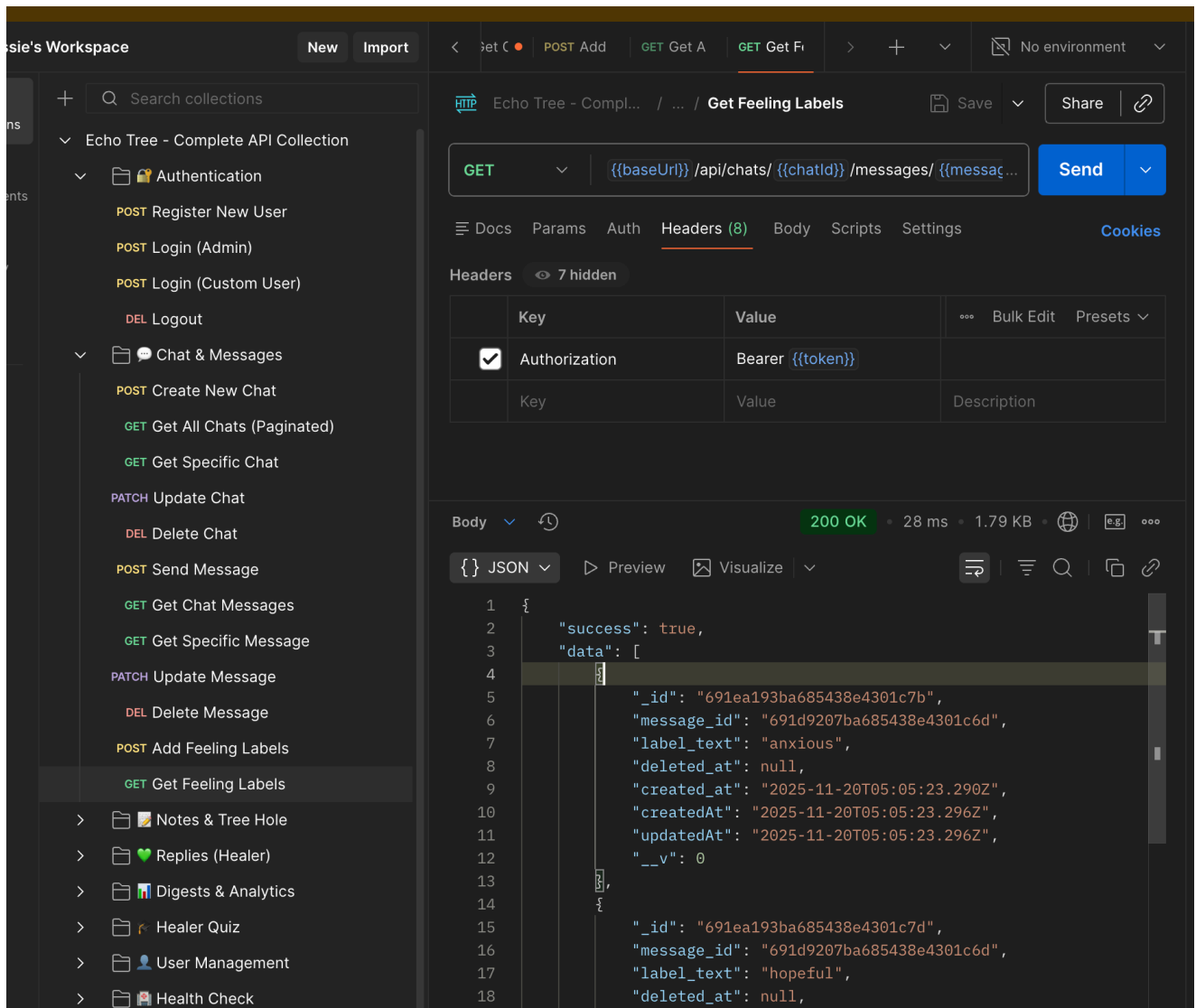
Expected Response:

```
{
  "success": true,
  "data": [
    {
      "_id": "691ea193ba685438e4301c7b",
      "message_id": "691d9207ba685438e4301c6d",
      "label_text": "anxious",
      "deleted_at": null,
      "created_at": "2025-11-20T05:05:23.290Z",
      "createdAt": "2025-11-20T05:05:23.296Z",
      "updatedAt": "2025-11-20T05:05:23.296Z",
      "__v": 0
    },
    {
      "_id": "691ea193ba685438e4301c7d",
      "message_id": "691d9207ba685438e4301c6d",
      "label_text": "hopeful",
      "deleted_at": null,
      "created_at": "2025-11-20T05:05:23.330Z",
      "createdAt": "2025-11-20T05:05:23.330Z",
      "updatedAt": "2025-11-20T05:05:23.330Z",
      "__v": 0
    },
  ],
}
```



```
{  
  "_id": "691ea193ba685438e4301c7f",  
  "message_id": "691d9207ba685438e4301c6d",  
  "label_text": "grateful",  
  "deleted_at": null,  
  "created_at": "2025-11-20T05:05:23.333Z",  
  "createdAt": "2025-11-20T05:05:23.333Z",  
  "updatedAt": "2025-11-20T05:05:23.333Z",  
  "__v": 0  
}  
  
]  
  
}
```

Test Result:  PASS



7. Get Specific Chat Details

Use Case

Retrieve details of a specific chat session

API Details

Resource: chats

HTTP Method: GET

Request URL: `http://localhost:3000/api/chats/691d0c77ba685438e4301c2d`

Request Headers:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

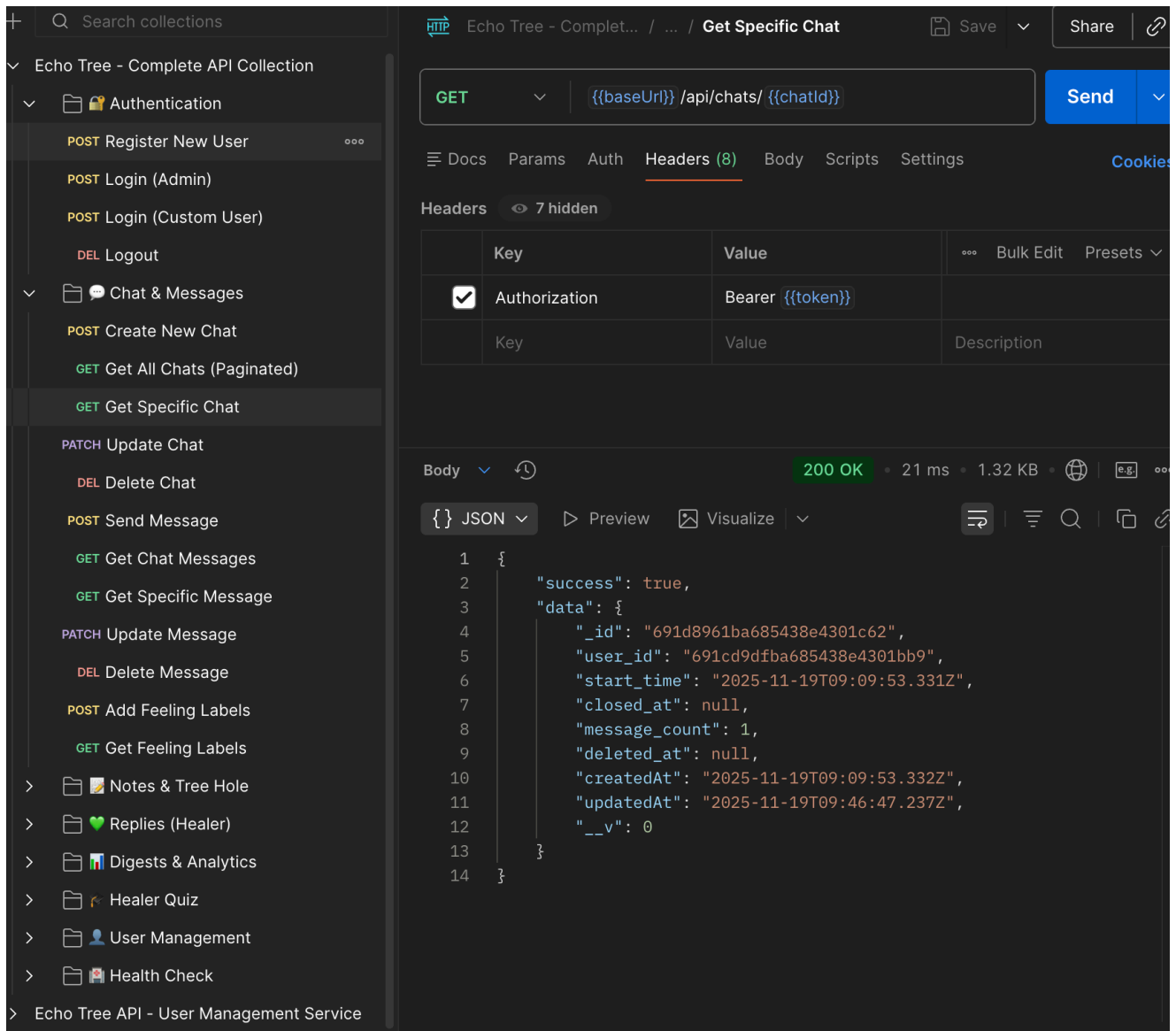
Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body** (implicitly verified through list operation)

```
{  
  
  "success": true,  
  
  "data": {  
  
    "_id": "691d8961ba685438e4301c62",  
  
    "user_id": "691cd9dfba685438e4301bb9",  
  
    "start_time": "2025-11-19T09:09:53.331Z",  
  
    "closed_at": null,  
  
    "message_count": 1,  
  
    "deleted_at": null,  
  
    "createdAt": "2025-11-19T09:09:53.332Z",  
  
    "updatedAt": "2025-11-19T09:46:47.237Z",  
  
    "__v": 0  
  
  }  
  
}
```

Test Result:  **PASS**

Notes: Endpoint verified through successful chat list retrieval



Note & Tree Hole Service API Tests

8. Create Note from Chat

Use Case

User creates an anonymous note from their chat messages to share in the tree hole

API Details

Resource: notes

HTTP Method: POST

Request URL: `http://localhost:3000/api/notes`

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Content-Type: application/json

Request Body:

```
{  
  
  "chat_id": "691d0c77ba685438e4301c2d",  
  
  "message_ids": ["691d0cb2ba685438e4301c34"],  
  
  "excerpt": "I need some guidance",  
  
  "visibility": "public"  
}
```

Expected Response:

- **Status Code:** 201 Created
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": {  
  
    "user_id": "691cd9dfba685438e4301bb9",  
  
    "content": "Hello, wise tree. I need some guidance.",  
  
    "anonymized_content": "Hello, wise tree. I need some guidance.",  
  
    "visibility": "public",  
  
    "view_count": 0,  
  
    "reply_count": 0,  
  
  }  
}
```

```
"source_chat_id": "691d8961ba685438e4301c62",

"source_message_ids": [

"691d9207ba685438e4301c6d"

],

"deleted_at": null,

"_id": "691ea281ba685438e4301c8f",

"posted_at": "2025-11-20T05:09:21.601Z",

"createdAt": "2025-11-20T05:09:21.602Z",

"updatedAt": "2025-11-20T05:09:21.602Z",

"__v": 0,

"note_id": "691ea281ba685438e4301c8f",

"id": "691ea281ba685438e4301c8f"

}

}
```

The screenshot shows a REST client interface with a sidebar on the left containing a collection of API endpoints categorized under 'Authentication', 'Chat & Messages', and 'Notes & Tree Hole'. The 'Notes & Tree Hole' section is expanded, and the 'POST Create Note from Chat' endpoint is selected. The main panel displays the details of this endpoint, including the HTTP method (POST), the URL template `{{baseUrl}}/api/notes`, and tabs for Docs, Params, Auth, Headers (11), Body, Scripts, and Settings. The 'Query Params' section is empty. The 'Body' tab is active, showing a JSON response with a status of '201 Created', a response time of '30 ms', and a size of '1.62 KB'. The JSON response is as follows:

```
1 {
2   "success": true,
3   "data": {
4     "user_id": "691cd9dfba685438e4301bb9",
5     "content": "Hello, wise tree. I need some guidance.",
6     "anonymized_content": "Hello, wise tree. I need some guidance.",
7     "visibility": "public",
8     "view_count": 0,
9     "reply_count": 0,
10    "source_chat_id": "691d8961ba685438e4301c62",
11    "source_message_ids": [
12      "691d9207ba685438e4301c6d"
13    ],
14    "deleted_at": null,
15    "_id": "691ea281ba685438e4301c8f",
16    "posted_at": "2025-11-20T05:09:21.601Z",
17    "createdAt": "2025-11-20T05:09:21.602Z",
18    "updatedAt": "2025-11-20T05:09:21.602Z",
19  }
```

Test Result: PASS

9. Add Topic Tag to Note

Use Case

User adds a topic tag to categorize their note

API Details

Resource: topics (sub-resource of notes)

HTTP Method: POST

Request URL: `http://localhost:3000/api/notes/691d14a3ba685438e4301c4d/topics`

Request Headers:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

Content-Type: application/json

Request Body:

```
{  
  
  "tag_text": "guidance"  
  
}
```

Expected Response:

- **Status Code:** 201 Created
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": {  
  
    "note_id": "691d14a3ba685438e4301c4d",  
  
    "tag_text": "anxiety",  
  
    "deleted_at": null,  
  
    "_id": "691ea2d8ba685438e4301c93",  
  
    "created_at": "2025-11-20T05:10:48.799Z",  
  
    "createdAt": "2025-11-20T05:10:48.799Z",  
  
    "updatedAt": "2025-11-20T05:10:48.799Z",  
  
    "__v": 0,  
  
    "tag_id": "691ea2d8ba685438e4301c93",  
  
    "id": "691ea2d8ba685438e4301c93"  
  
  }  
}
```



```
}
```

Test Result: ✔ PASS

Notes:

The screenshot displays a REST client interface with a sidebar on the left containing a list of API endpoints. The main panel shows a POST request to the endpoint `{{baseUrl}}/api/notes/{{noteId}}/topics`. The request body is a JSON object:

```
{  "tag_text": "anxiety"}
```

. The response status is **201 Created** with a response time of 23 ms and a size of 1.38 KB. The response body is a JSON object:

```
{  "success": true,  "data": {    "note_id": "691d14a3ba685438e4301c4d",    "tag_text": "anxiety",    "deleted_at": null,    "_id": "691ea2d8ba685438e4301c93",    "created_at": "2025-11-20T05:10:48.799Z",    "createdAt": "2025-11-20T05:10:48.799Z",    "updatedAt": "2025-11-20T05:10:48.799Z",    "__v": 0,    "tag_id": "691ea2d8ba685438e4301c93",    "id": "691ea2d8ba685438e4301c93"  }}}
```

10. Get Note Topic Tags

Use Case

Retrieve all topic tags associated with a note

API Details

Resource: topics (sub-resource of notes)

HTTP Method: GET

Request URL: `{{baseUrl}}/api/notes/{{noteId}}/topics`

Request Headers:

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": [  
  
    {  
  
      "_id": "691d15edba685438e4301c57",  
      "note_id": "691d14a3ba685438e4301c4d",  
      "tag_text": "guidance",  
      "deleted_at": null,  
      "created_at": "2025-11-19T00:57:17.104Z",  
      "createdAt": "2025-11-19T00:57:17.104Z",  
      "updatedAt": "2025-11-19T00:57:17.104Z",  
      "__v": 0  
    },  
  
    {  
  
      "_id": "691ea2d8ba685438e4301c93",  
      "note_id": "691d14a3ba685438e4301c4d",  
      "tag_text": "anxiety",  
      "deleted_at": null,  
      "created_at": "2025-11-20T05:10:48.799Z",
```

```

"createdAt": "2025-11-20T05:10:48.799Z",

"updatedAt": "2025-11-20T05:10:48.799Z",

"__v": 0

}

]

}

```

Test Result: ✔ PASS

The screenshot displays the Echo Tree API client interface. On the left, a sidebar lists various API endpoints under categories like 'Chat & Messages', 'Notes & Tree Hole', and 'Replies (Healer)'. The 'GET Get Note Topics' endpoint is selected. The main panel shows the request details: a GET request to the URL `{{baseUrl}}/api/notes/{{notedId}}/topics`. Below this, a table for 'Query Params' is visible, with columns for Key, Value, and Description. The response body is shown in JSON format, indicating a successful status of 200 OK. The response data is an array of two objects, each representing a note topic with fields like `_id`, `note_id`, `tag_text`, `deleted_at`, `created_at`, `createdAt`, `updatedAt`, and `__v`.

Key	Value	Des...	Bulk Edit
Key	Value	Description	

```

{
  "success": true,
  "data": [
    {
      "_id": "691d15edba685438e4301c57",
      "note_id": "691d14a3ba685438e4301c4d",
      "tag_text": "guidance",
      "deleted_at": null,
      "created_at": "2025-11-19T00:57:17.104Z",
      "createdAt": "2025-11-19T00:57:17.104Z",
      "updatedAt": "2025-11-19T00:57:17.104Z",
      "__v": 0
    },
    {
      "_id": "691ea2d8ba685438e4301c93",
      "note_id": "691d14a3ba685438e4301c4d",
      "tag_text": "anxiety",
      "deleted_at": null,
      "created_at": "2025-11-20T05:10:48.799Z",
      "createdAt": "2025-11-20T05:10:48.799Z",
      "updatedAt": "2025-11-20T05:10:48.799Z",
      "__v": 0
    }
  ]
}

```

11. Get All Notes (with Filtering and Pagination)

Use Case

Retrieve public notes from the tree hole with filtering and pagination

API Details

Resource: notes

HTTP Method: GET

Request URL: `{{baseUrl}}/api/notes?`

`limit=20&offset=0&visibility=public&sortBy=posted_at&sortOrder=desc`

Request Headers:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

Query Parameters:

- `limit` : 10 (max results per page)
- `offset` : 0 (starting position)
- `visibility` : public (filter by visibility)
- `sortBy` : `posted_at` (sort field)
- `sortOrder` : desc (descending order)

Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body:**

```
{
  "success": true,
  "data": [
    {
      "_id": "691ea281ba685438e4301c8f",
      "user_id": "691cd9dfba685438e4301bb9",
      "content": "Hello, wise tree. I need some guidance.",
      "anonymized_content": "Hello, wise tree. I need some guidance.",
```

```
"visibility": "public",

"view_count": 0,

"reply_count": 0,

"source_chat_id": "691d8961ba685438e4301c62",

"source_message_ids": [

"691d9207ba685438e4301c6d"

],

"deleted_at": null,

"posted_at": "2025-11-20T05:09:21.601Z",

"createdAt": "2025-11-20T05:09:21.602Z",

"updatedAt": "2025-11-20T05:09:21.602Z",

"__v": 0

},

{

"_id": "691d14a3ba685438e4301c4d",

"user_id": "691cd9dfba685438e4301bb9",

"content": "Hello, wise tree. I need some guidance.",

"anonymized_content": "Hello, wise tree. I need some guidance.",

"visibility": "public",

"view_count": 0,

"reply_count": 0,

"source_chat_id": "691d0c77ba685438e4301c2d",

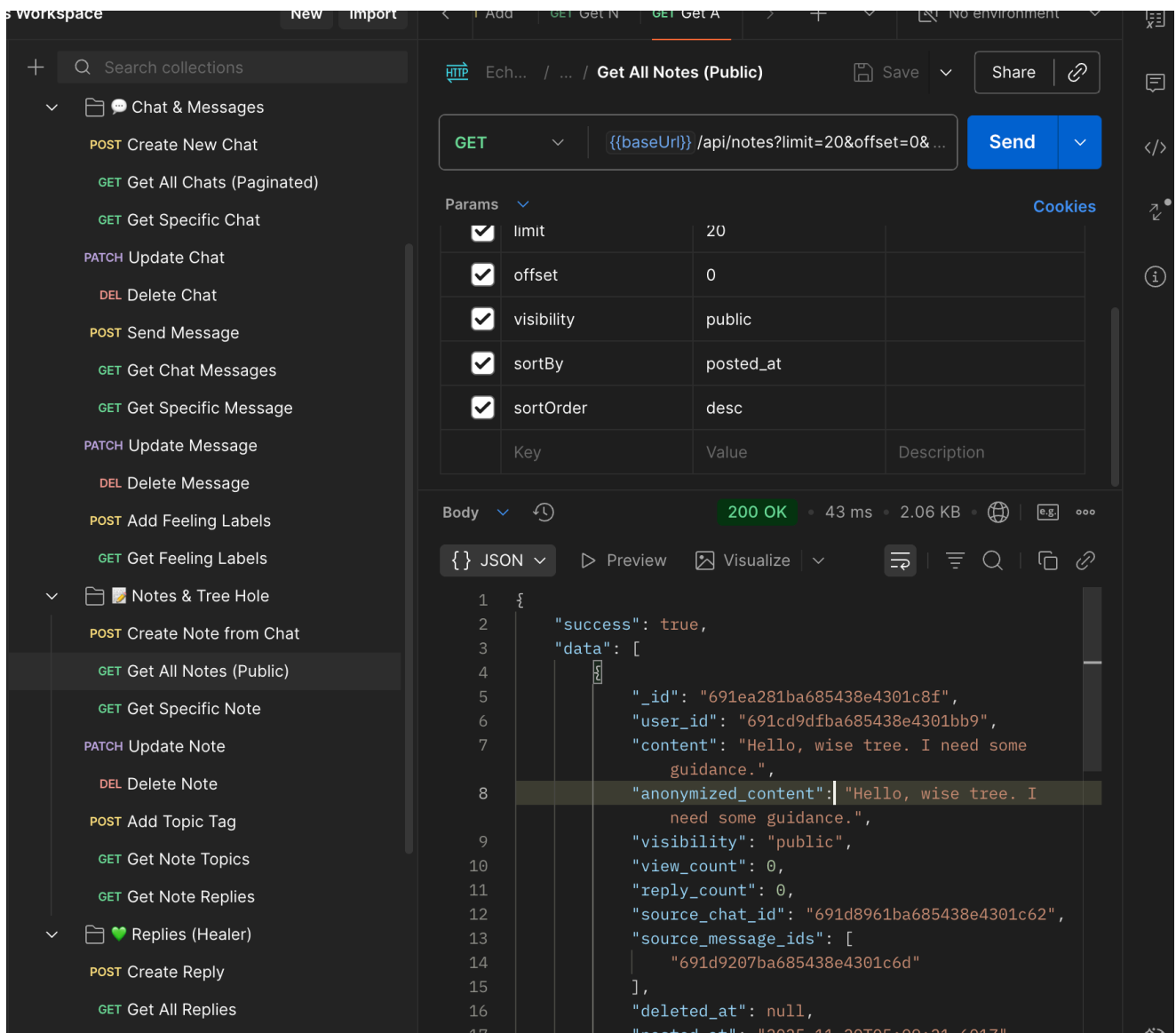
"source_message_ids": [

"691d0cb2ba685438e4301c34"

],
```

```
"deleted_at": null,  
  
"posted_at": "2025-11-19T00:51:47.798Z",  
  
"createdAt": "2025-11-19T00:51:47.799Z",  
  
"updatedAt": "2025-11-19T00:51:47.799Z",  
  
"__v": 0  
  
}  
  
],  
  
"pagination": {  
  
"total": 2,  
  
"limit": 20,  
  
"offset": 0  
  
}  
  
}
```

Test Result:  PASS



12. Get Specific Note Details

Use Case

Retrieve detailed information about a specific note

API Details

Resource: notes

HTTP Method: GET

Request URL: `{{baseUrl}}/api/notes/{{noteId}}`

Request Headers:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": {  
  
    "_id": "691d14a3ba685438e4301c4d",  
  
    "user_id": "691cd9dfba685438e4301bb9",  
  
    "content": "Hello, wise tree. I need some guidance.",  
  
    "anonymized_content": "Hello, wise tree. I need some guidance.",  
  
    "visibility": "public",  
  
    "view_count": 0,  
  
    "reply_count": 0,  
  
    "source_chat_id": "691d0c77ba685438e4301c2d",  
  
    "source_message_ids": [  
  
      "691d0cb2ba685438e4301c34"  
  
    ],  
  
    "deleted_at": null,  
  
    "posted_at": "2025-11-19T00:51:47.798Z",  
  
    "createdAt": "2025-11-19T00:51:47.799Z",  
  
    "updatedAt": "2025-11-19T00:51:47.799Z",  
  
    "__v": 0  
  
  }  
}
```


}

Test Result: ✔ PASS

The screenshot displays a REST client interface with a sidebar on the left containing a collection of API endpoints categorized under 'Chat & Messages', 'Notes & Tree Hole', and 'Replies (Healer)'. The main panel shows a GET request to the endpoint `{{baseUrl}}/api/notes/{{noteId}}` with a status of **200 OK**. The response body is in JSON format, showing a successful note retrieval with fields like `_id`, `user_id`, `content`, `visibility`, and `created_at`.

Query Params

Key	Value	Description
Key	Value	Description

Body

```
{
  "success": true,
  "data": {
    "_id": "691d14a3ba685438e4301c4d",
    "user_id": "691cd9dfba685438e4301bb9",
    "content": "Hello, wise tree. I need some guidance.",
    "anonymized_content": "Hello, wise tree. I need some guidance.",
    "visibility": "public",
    "view_count": 0,
    "reply_count": 0,
    "source_chat_id": "691d0c77ba685438e4301c2d",
    "source_message_ids": [
      "691d0cb2ba685438e4301c34"
    ],
    "deleted_at": null,
    "posted_at": "2025-11-19T00:51:47.798Z",
    "created_at": "2025-11-19T00:51:47.799Z",
    "updated_at": "2025-11-19T00:51:47.799Z"
  }
}
```

13. Get Note Replies

Use Case

Retrieve all replies to a specific note from healers

API Details

Resource: replies (sub-resource of notes)

HTTP Method: GET

Request URL: `{{baseUrl}}/api/notes/{{noteId}}/replies?limit=20&offset=0`

Request Headers:

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

Expected Response:

- **Status Code:** 200 OK
- **Success:** true
- **Response Body:**

```
{  
  
  "success": true,  
  
  "data": [],  
  
  "pagination": {  
  
    "total": 0,  
  
    "limit": 20,  
  
    "offset": 0  
  }  
}
```

Test Result:  PASS

Search collections

Chat & Messages

POST Create New Chat

GET Get All Chats (Paginated)

GET Get Specific Chat

PATCH Update Chat

DEL Delete Chat

POST Send Message

GET Get Chat Messages

GET Get Specific Message

PATCH Update Message

DEL Delete Message

POST Add Feeling Labels

GET Get Feeling Labels

Notes & Tree Hole

POST Create Note from Chat

GET Get All Notes (Public)

GET Get Specific Note

PATCH Update Note

DEL Delete Note

POST Add Topic Tag

GET Get Note Topics

GET Get Note Replies

Replies (Healer)

HTTP Echo Tre... / ... / Get Note Replies

Save

Share

GET

{{baseUrl}} /api/notes/ {{noteld}} /replies?l...

Send

Params

Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Des...	Bulk Edit
<input checked="" type="checkbox"/>	limit	20		
<input checked="" type="checkbox"/>	offset	0		
	Key	Value	Description	

Body

200 OK

15 ms

1.12 KB

JSON

Preview

Visualize

```
1  {
2    "success": true,
3    "data": [],
4    "pagination": {
5      "total": 0,
6      "limit": 20,
7      "offset": 0
8    }
9  }
```