Time Capsule API — Testing Guide

Manual testing

This section contains the full list of endpoints with sample requests, headers, and expected behaviours for testing via API clients like **Thunder Client**, or **Postman**.

Authentication

1. Register User

- **Method:** POST
- **URL:** /auth/register

Headers:

Content-Type: application/json

Body (raw JSON):

```
{
"email": "user@example.com",
"password": "yourpassword"
}
```

Expected Response:

- 201 Created if successful
- 400 if user already exists

Manual Testing Output:

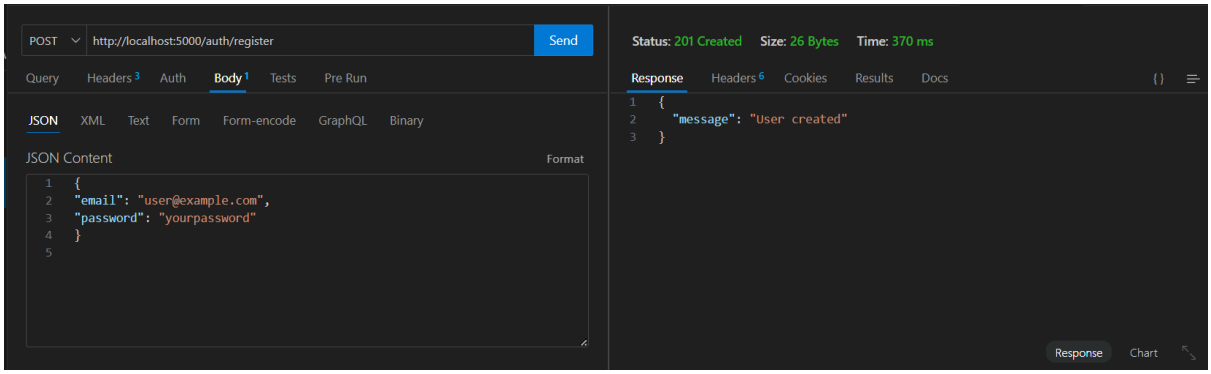

Figure 1: Successful user registration

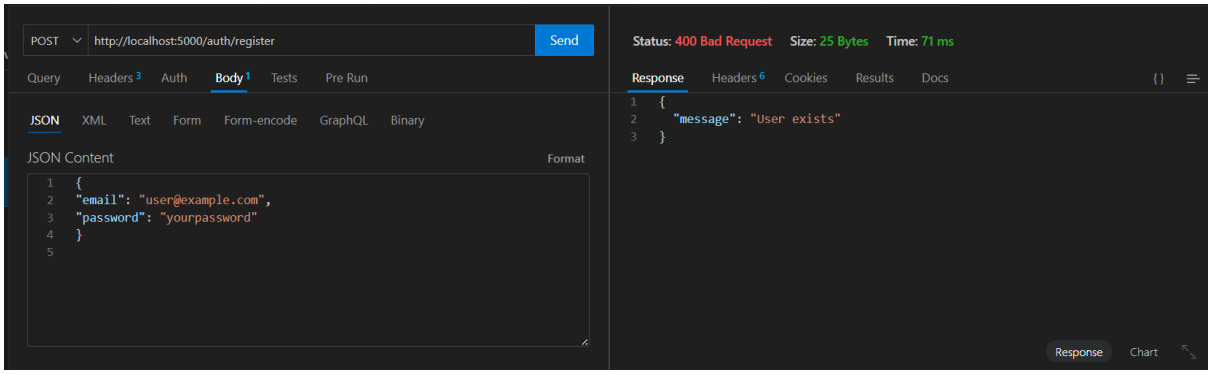

Figure 2: 400 error; user already exists

2. Login User

- **Method:** POST
- **URL:** /auth/login

Body:

```
{
"email": "user@example.com",
"password": "yourpassword"
}
```

Expected Response:

```
{
  "token": "<JWT_TOKEN>"
}
```

Manual Testing Output:

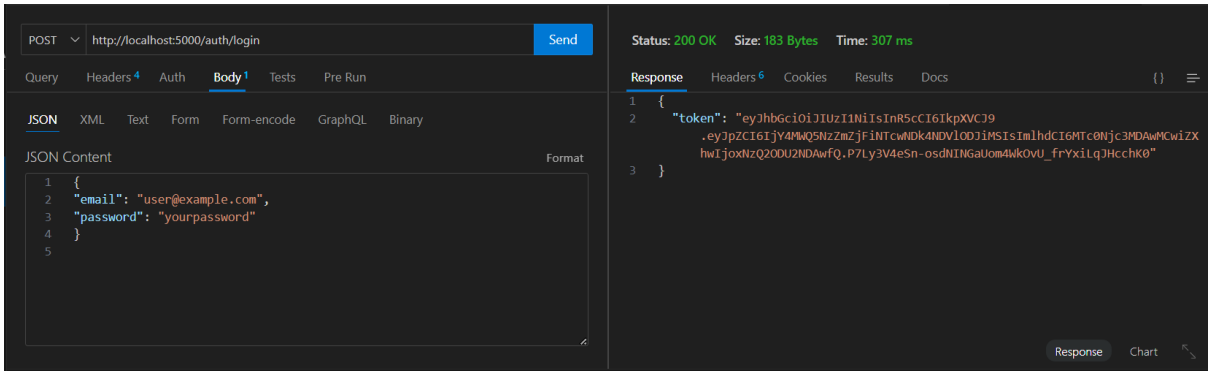

Figure 3: JWT Token is visible

Capsule Endpoints

All capsule endpoints require the header:

Authorization: Bearer <JWT_TOKEN>

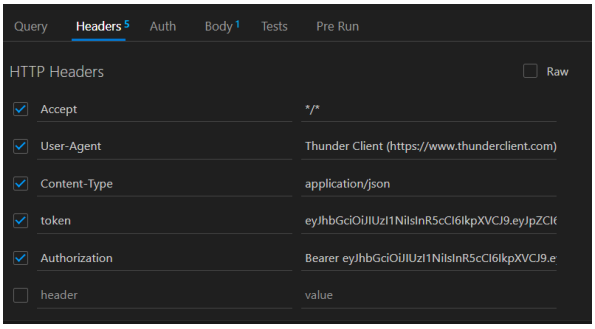

Figure 4: Require Headers

Create Capsule

- **Method:** POST
- **URL:** /capsules

Body:

```
{
 "message": "Hello future!",
 "unlock_at": "2025-12-31T23:59:59Z"
}
```

Expected Response:

```
{
 "id": "<CAPSULE_ID>",
 "unlock_code": "abc12345"
}
```

Manual Testing Output:

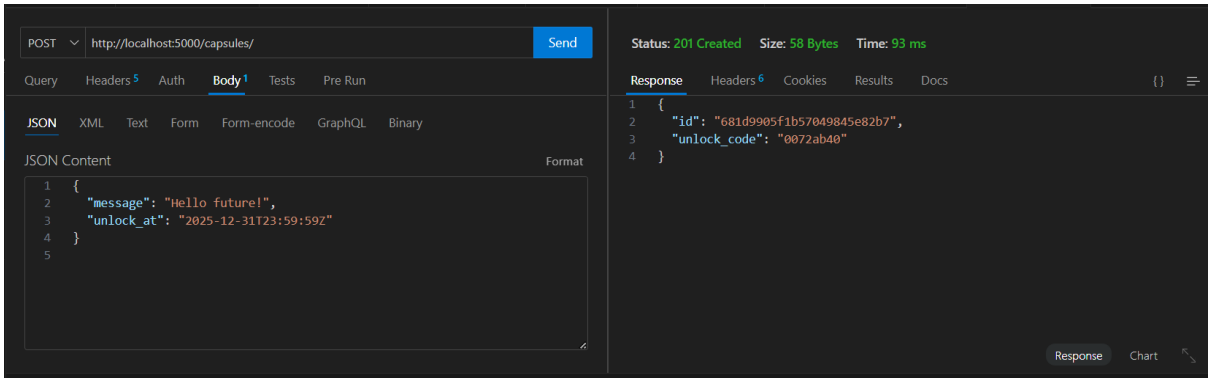

Figure 5: Capsule Created Successfully

Get Capsule (By ID & Unlock Code)

- **Method:** GET
- **URL:** /capsules/:id?code=<UNLOCK_CODE>

Expected Responses:

- 200 OK → Returns message if unlocked
- 401 Unauthorized → Wrong/missing code
- 403 Forbidden → Capsule still locked

- 410 Gone → Capsule expired

Manual Testing Output:

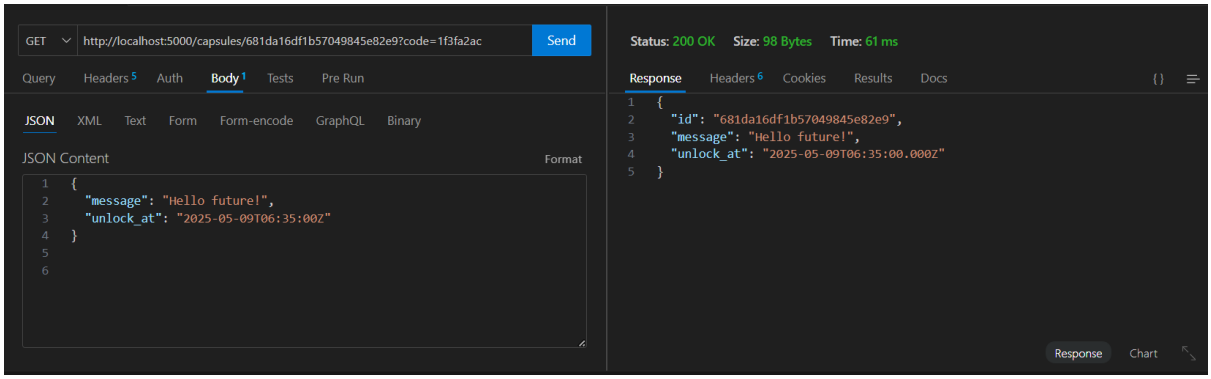

Figure 6:200 OK; Unlocked Capsule

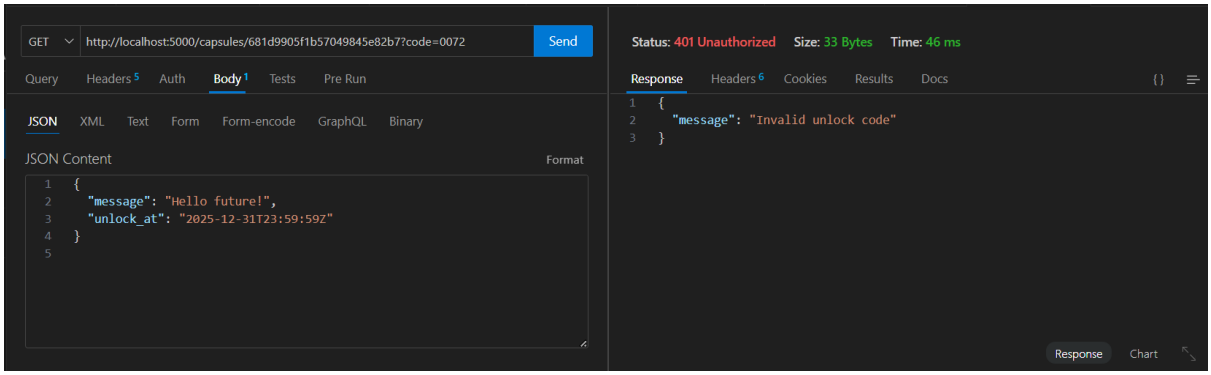

Figure 7: 401 Unauthorized; Wrong Code

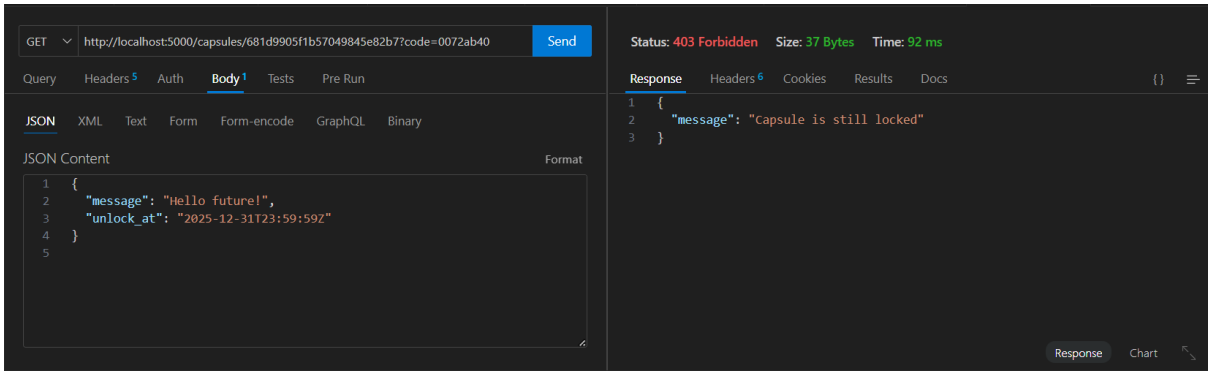

Figure 8: 403 Forbidden; Capsule is still locked

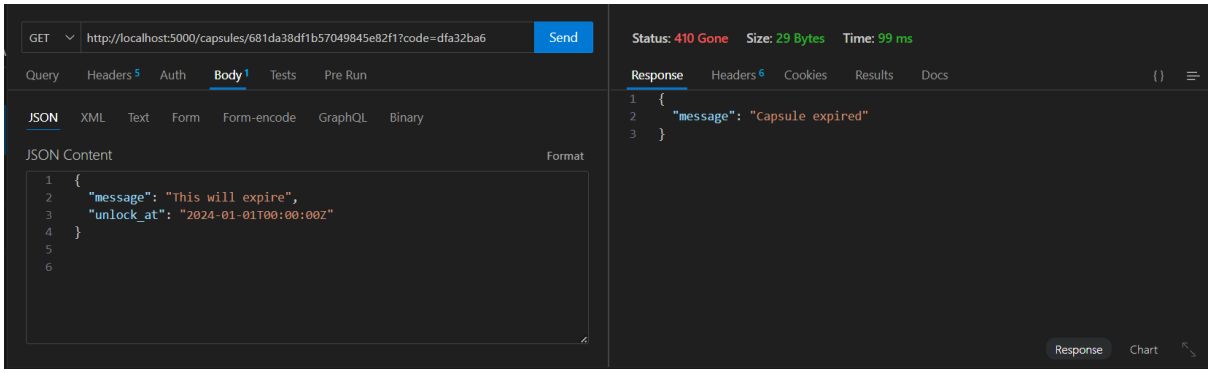

Figure 9: 401 Gone; Capsule is expired

List Capsules (Paginated)

- **Method:** GET
- **URL:** /capsules?page=1&limit=5

Expected Response:

```
{
 "page": 1,
 "limit": 5,
 "capsules": [
  {
   "id": "...",
   "unlock_at": "...",
   "created_at": "...",
   "message": "..." // Only if unlocked
  }
 ]
}
```

Manual Testing Output:

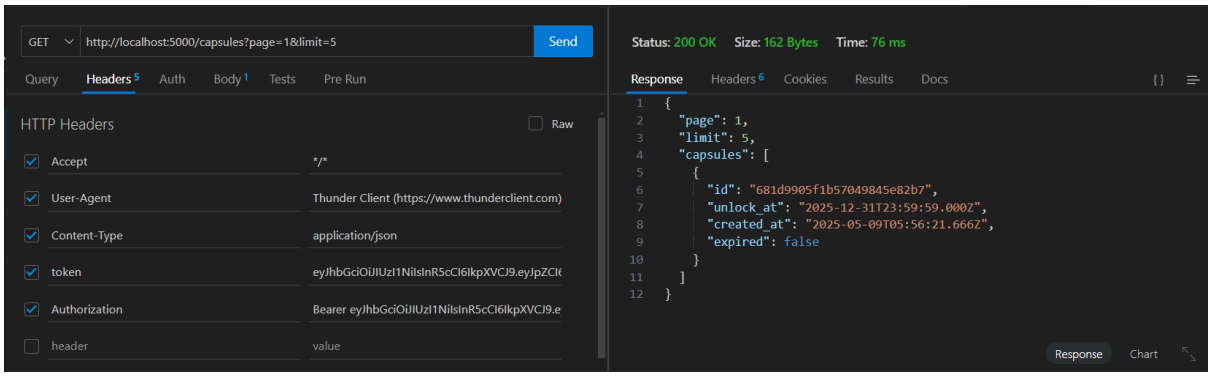

Figure 10: List of capsules

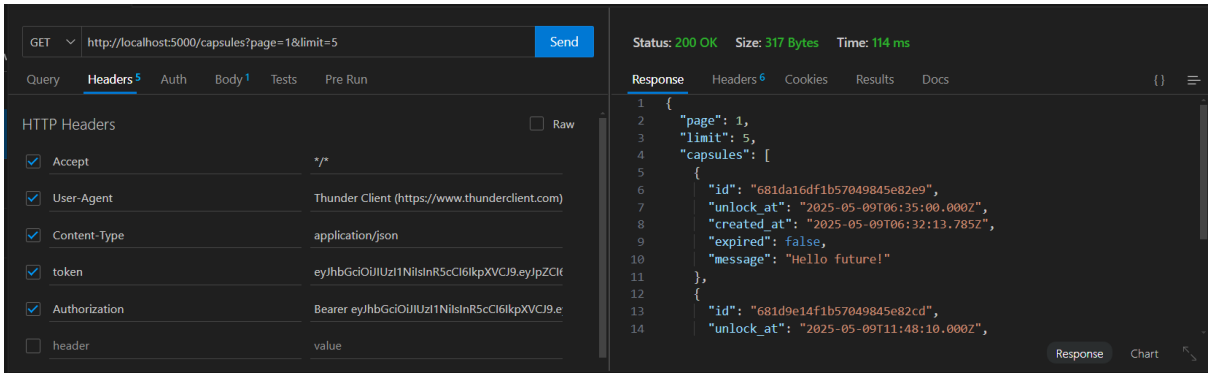

Figure 11: List of capsules and an unlocked capsule with the message

Update Capsule

- **Method:** PUT
- **URL:** /capsules/:id?code=<UNLOCK_CODE>

Body:

```
{
 "message": "Updated message",
```

"unlock_at": "2026-01-01T00:00:00Z"
}

Expected Responses:

- 200 OK → Success
- 401 → Invalid unlock code
- 403 → Capsule already unlocked

Manual Testing Output:

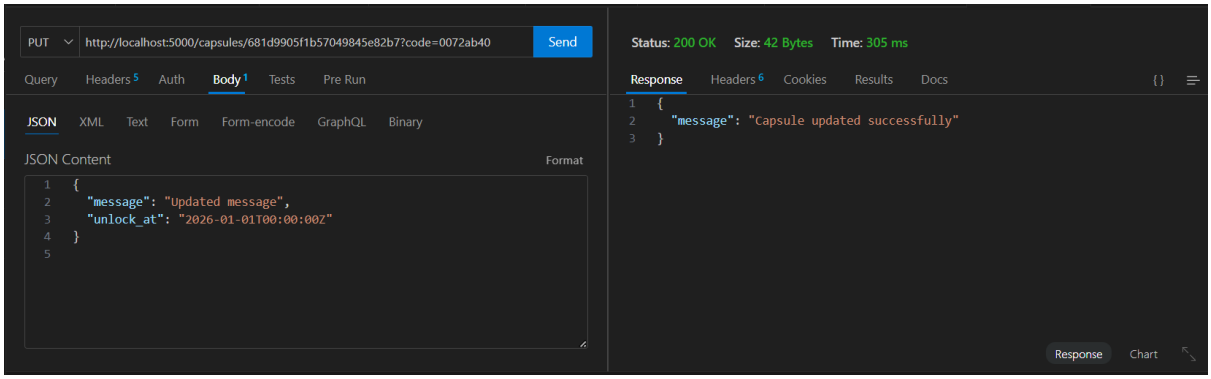

Figure 12: 200 OK; Capsule updated successfully

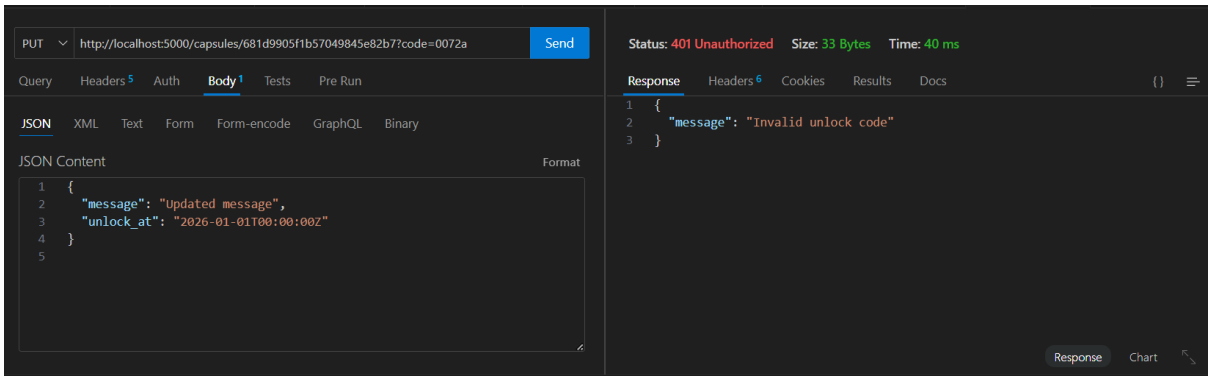

Figure 13; 401 Unauthorized; Wrong unlock code

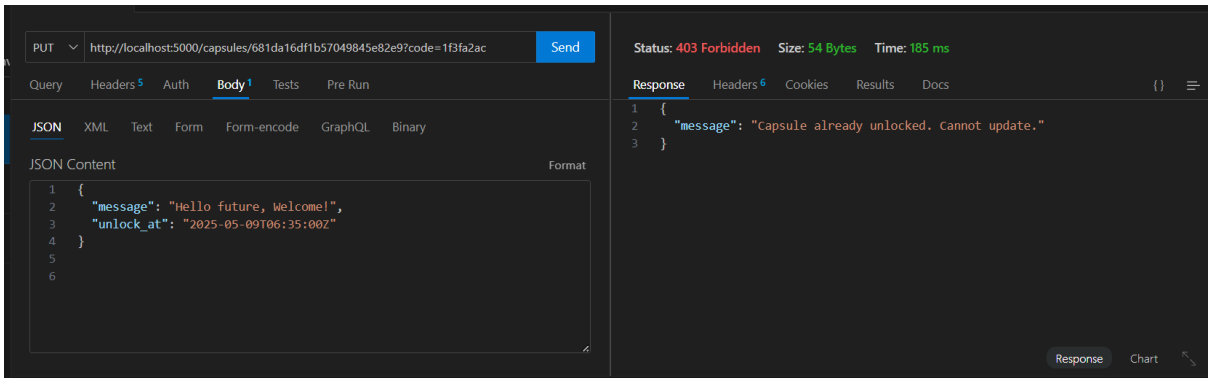

Figure 14: 403 Forbidden; Capsule already unlocked, thus cannot be updated

Delete Capsule

- **Method:** DELETE
- **URL:** /capsules/:id?code=<UNLOCK_CODE>

Expected Responses:

- 200 OK → Capsule deleted
- 401 → Wrong code
- 403 → Unlock time has passed
- 404 → Already deleted or wrong ID

Manual Testing Output:

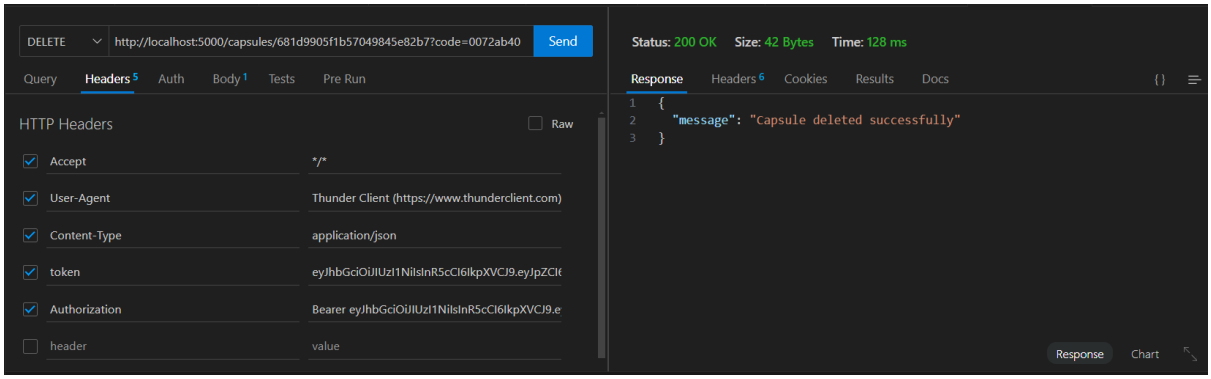

Figure 15: 200 OK; Capsule deleted successfully

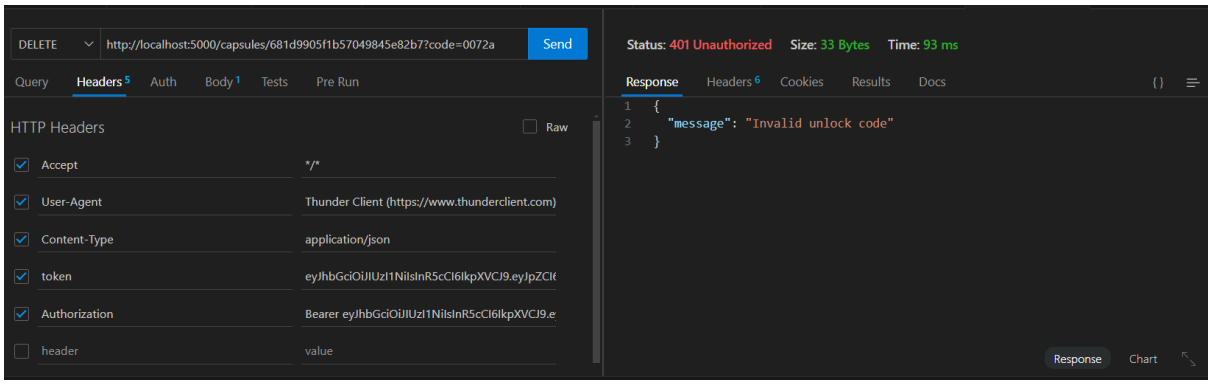

Figure 16: 401 Unauthorized; Wrong unlock code

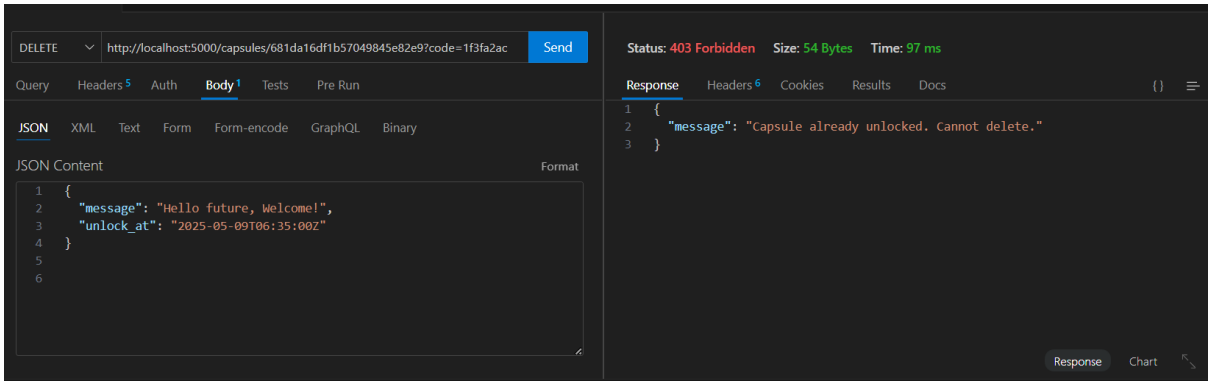

Figure 17:403 Forbidden; Capsule already unlocked

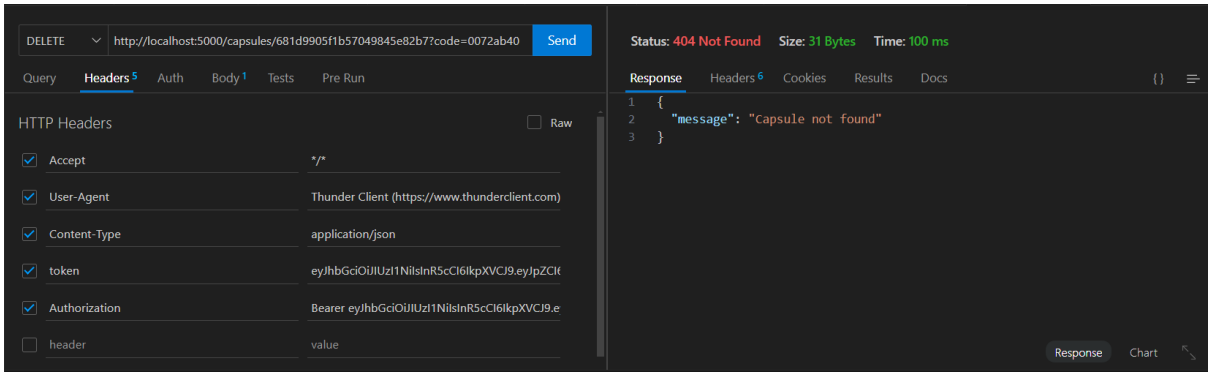

Figure 18: 404 Not found; Capsule already deleted

Expiration Behaviour

- Capsules **expire automatically** 30 days after unlock_at.
- If GET /capsules/:id is called for an expired capsule, response is:

```
{
  "message": "Capsule expired"
}
```

Manual Testing Output:

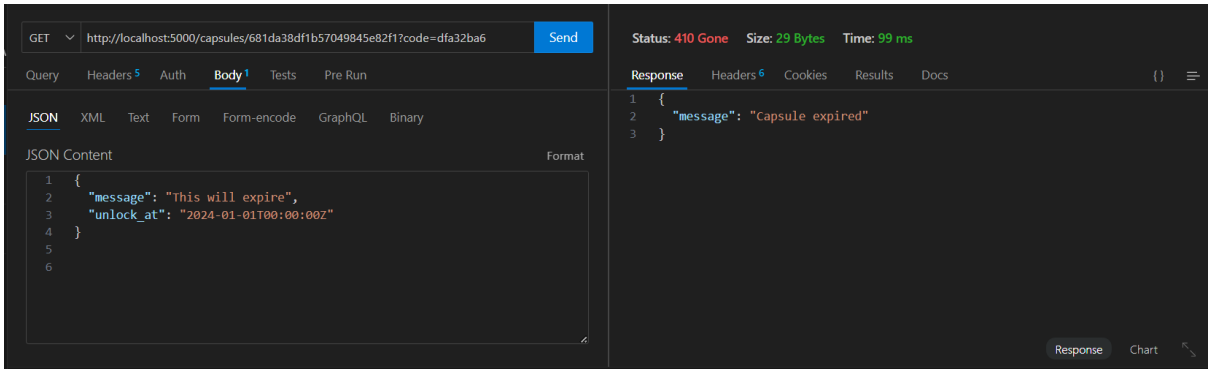

Figure 19: 410 Gone; Capsule Expired

Jest Testing

Setup Instructions

PowerShell command:

1. Install Dev Dependencies: npm install --save-dev jest supertest
2. To run the tests: npm test

auth.test.js

1. **User Registration**
   - Registers a new user with a unique email.
   - Expects:
     - 201 Created
     - message: "User created"
2. **User Login**
   - Logs in with registered credentials.
   - Expects:
     - 200 OK
     - A valid JWT token in response.

capsule.test.js

1. **Setup**
   - Creates a test user and logs in.
   - Reuses the token in all capsule operations.
   - Clears relevant collections before/after tests.

2. **Capsule Creation**
   - Endpoint: POST /capsules
   - Creates a new capsule with message and unlock_at.
   - Expects:
     - 201 Created
     - Returns capsule id and unlock_code.

3. **Capsule Retrieval (GET)**

- Endpoint: GET /capsules/:id?code=...

Tested Scenarios:

Condition	Expected Status	Behavior
Valid, still locked	403	Capsule is locked
Valid, wrong unlock code	401	Unauthorized
Capsule expired	410	Marked as expired

4. **Capsule Update (PUT)**

- Endpoint: PUT /capsules/:id?code=...

Tested Scenarios:

Condition	Expected Status	Behavior
Valid code, still locked	200	Capsule updated
Wrong unlock code	401	Invalid unlock code
After unlock time	403	Forbidden to update

5. **Capsule Delete (DELETE)**

- Endpoint: DELETE /capsules/:id?code=...

Tested Scenarios:

Condition	Expected Status	Behavior
Valid, still locked	200	Capsule deleted
Already deleted	404	Capsule not found

6. **Capsule Listing (Pagination)**

- Endpoint: GET /capsules?page=1&limit=5

Tested Scenarios:

Condition	Expected Status	Behavior

Page 1 request	200	Returns up to 5 capsules
Page 2 request	200	Returns next batch

7. Expiration Logic

- Manually marks a capsule as is_expired: true in the DB.
- Requests the capsule and expects:
  - 410 Gone
  - "Capsule expired" message

Testing Output:

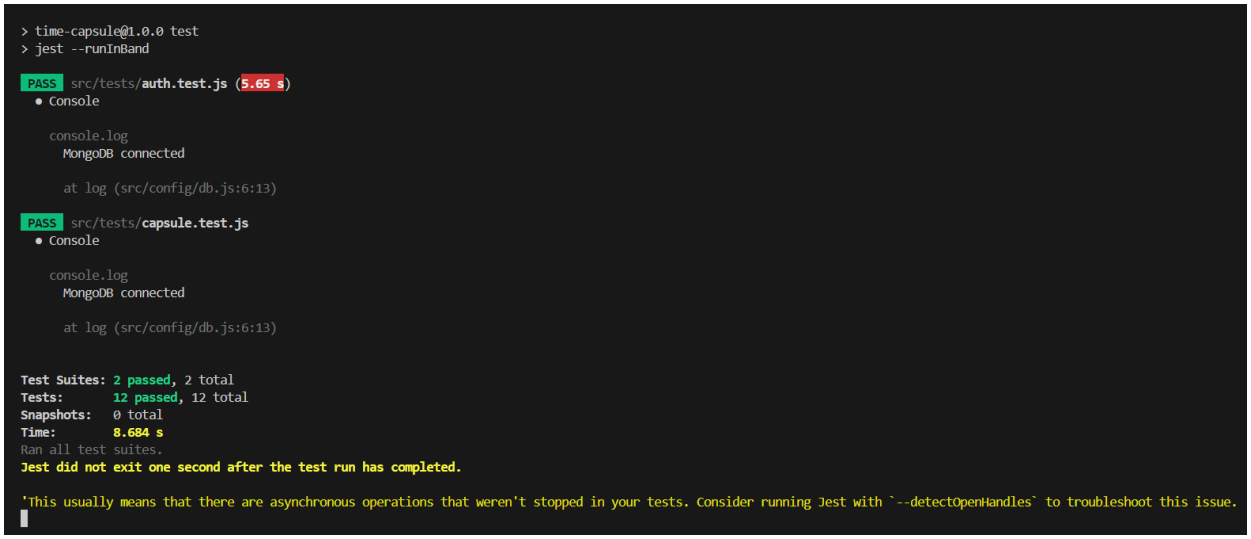

Figure 20: Test cases passed for all the functionality

```
PS E:\Internship\Guidespoint\Time Capsule> npm test -- --verbose

> time-capsule@1.0.0 test
> jest --runInBand --verbose

  console.log
    MongoDB connected

      at log (src/config/db.js:6:13)

 PASS  src/tests/auth.test.js
  Auth Tests
    ✓ should register a new user (1512 ms)
    ✓ should login and return a token (409 ms)

  console.log
    MongoDB connected

      at log (src/config/db.js:6:13)

 PASS  src/tests/capsule.test.js
  Capsule Creation
    ✓ should create a new capsule (51 ms)
  Capsule Retrieval with Unlock Logic
    ✓ should return 403 if unlock time has not passed (60 ms)
    ✓ should return 401 for invalid unlock code (35 ms)
  Capsule Pagination
    ✓ should return first page of capsules (41 ms)
    ✓ should return second page of capsules (56 ms)
  Capsule Update
    ✓ should update the capsule before unlock time (102 ms)
    ✓ should return 401 with wrong unlock code (52 ms)
  Capsule Delete
    ✓ should delete the capsule before unlock time (84 ms)
    ✓ should return 404 when trying to delete again (49 ms)
  Expired Capsule Retrieval
    ✓ should return 410 for expired capsule (41 ms)

Test Suites: 2 passed, 2 total
Tests:       12 passed, 12 total
Snapshots:   0 total
Time:        6.194 s, estimated 9 s
Ran all test suites.
Jest did not exit one second after the test run has completed.

'This usually means that there are asynchronous operations that wer
```

Figure 21: Detailed report of all the passed test cases