Software Quality Assurance Fundamentals Final Project Report

Comprehensive Testing of MockAPI Application

Group 7 Members**:**

Mohammed Sadi Mahmoud Nemer (2020089)

Mohmad Moaena (2019759)

Rayan Abu Gharbieh (2100495)

Qamar Mourad (2104729)

Submitted To:

Prof. Hilal Rakıcı

COP4601 param.com - Fundamentals of Software Quality Assurance

December 24, 2024

Introduction

This report outlines the comprehensive testing of the MockAPI application as part of the Software Quality Assurance Fundamentals course. The project involved API testing using Postman and UI testing using Selenium to validate functionality, performance, and reliability.

Key achievements include:

- Execution of 8 API test cases covering CRUD operations.
- Performance and reliability tests revealing consistent application behavior under stress.
- Automation of UI workflows, including login, resource management, and modal validation.
- Detailed documentation of successes, challenges, and recommendations for future improvements.

1. Application and API Analysis

MockAPI is a platform designed to simulate RESTful APIs and provide a user-friendly interface for testing and managing resources. It offers endpoints for CRUD operations, making it suitable for API and UI testing.

Functional Areas Tested

- **Authentication:** User login validation with valid and invalid credentials.
- **Resource Management:** Adding, updating, and deleting resources.
- **Navigation:** Verifying page transitions between dashboard, projects, and resources.
- **Performance:** Evaluating API reliability and stress-handling capabilities.

2. Test Cases Creation

Both positive and negative test scenarios were developed for API and UI testing, covering the following areas:

2.1. API Test Cases

Test Case ID	Description	HTTP Method	Endpoint	Expected Outcome	Actual Outcome
API-1	Get all tasks	GET	/api/project/tasks	200 OK, array of tasks	Passed
API-2	Get non-existent task	GET	/api/project/tasks/{id}	404 Not Found	Passed
API-3	Create a new task	POST	/api/project/tasks	201 Created	Passed
API-4	Invalid URL	POST	/invalid-url	400 Bad Request	Passed
API-5	Update an existing task	PUT	/api/project/tasks/{id}	200 OK, updated task	Passed
API-6	Update non-existent task	PUT	/api/project/tasks/{id}	404 Not Found	Passed
API-7	Delete an existing task	DELETE	/api/project/tasks/{id}	200 OK, task deleted	Passed
API-8	Delete non-existent task	DELETE	/api/project/tasks/{id}	404 Not Found	Passed

2.2. Performance and Reliability Tests

Simulated Load Test

- **Description**: Executed 100 GET and POST requests with minimal delay of 10ms.
- **Result**: Majority of requests returned 200 OK, but some resulted in "429 Too Many Requests." and " Over rate limit."

Stress Test

- **Description**: Created 100 tasks rapidly.
- **Result**: Some requests failed due to server overload, resulting in "429 Too Many Requests."

Reliability Test

- **Description**: Sent repeated GET requests to test response consistency.
- **Result**: Application maintained stability under consistent load.

2.3. UI Test Cases

Test Case ID	Description	Steps	Expected Outcome	Actual Outcome
UI-1	Valid login	Enter valid credentials and click login	Redirects to Projects page	Passed
UI-2	Invalid login	Enter invalid credentials and click login	Displays error message	Passed
UI-3	Add a new resource	Click "New Resource", enter valid data, and click "Create"	Resource added successfully	Passed
UI-4	Edit an existing resource	Click "Edit", modify resource data, and click "Update"	Resource updated successfully	Passed
UI-5	Delete an existing resource	Click delete icon, confirm in modal by clicking "Delete"	Resource deleted successfully	Passed
UI-6	Verify navigation between sections	Navigate between Dashboard and Projects sections	Navigation works without issues	Passed
UI-7	Validate "Delete Resource" modal appearance	Trigger delete action and validate the modal title "Delete Resource"	Modal appears with correct title	Passed
UI-8	Validate button interactions	Click buttons (login, create, update, delete) and validate their behavior	Buttons behave as expected	Passed
UI-9	Performance test for resource interactions	Perform multiple resource add, edit, and delete operations under high-frequency interactions	Application remains responsive	Passed
UI-10	Validate error handling for invalid form submissions	Submit forms with invalid or incomplete data	Displays appropriate error messages	Passed

3. API Testing Results

3.1. Valid Scenarios:

- Successfully retrieved, created, updated, and deleted resources.
- Validated error handling for non-existent resources and invalid requests.

3.2. Performance and Reliability Tests:

Simulated Load Test:

- **Description:** 500 GET requests with minimal delay.
- **Result:** Majority returned 200 OK, with occasional "429 Too Many Requests" errors during high load.

Stress Test:

- **Description:** Rapid task creation (100 POST requests).
- **Result:** Some requests failed with "429 Too Many Requests" due to server overload.

Reliability Test:

- **Description:** Repeated GET requests to assess consistent behavior.
- **Result:** Application maintained stability.

---

4. UI Testing Results

4.1. Key Scenarios:

1. Valid login redirected users to the dashboard.
2. Invalid login displayed appropriate error messages.
3. Resources were successfully created, updated, and deleted.
4. Page transitions and modal interactions worked seamlessly.
5. Button clicks and user interactions were validated for proper functionality.

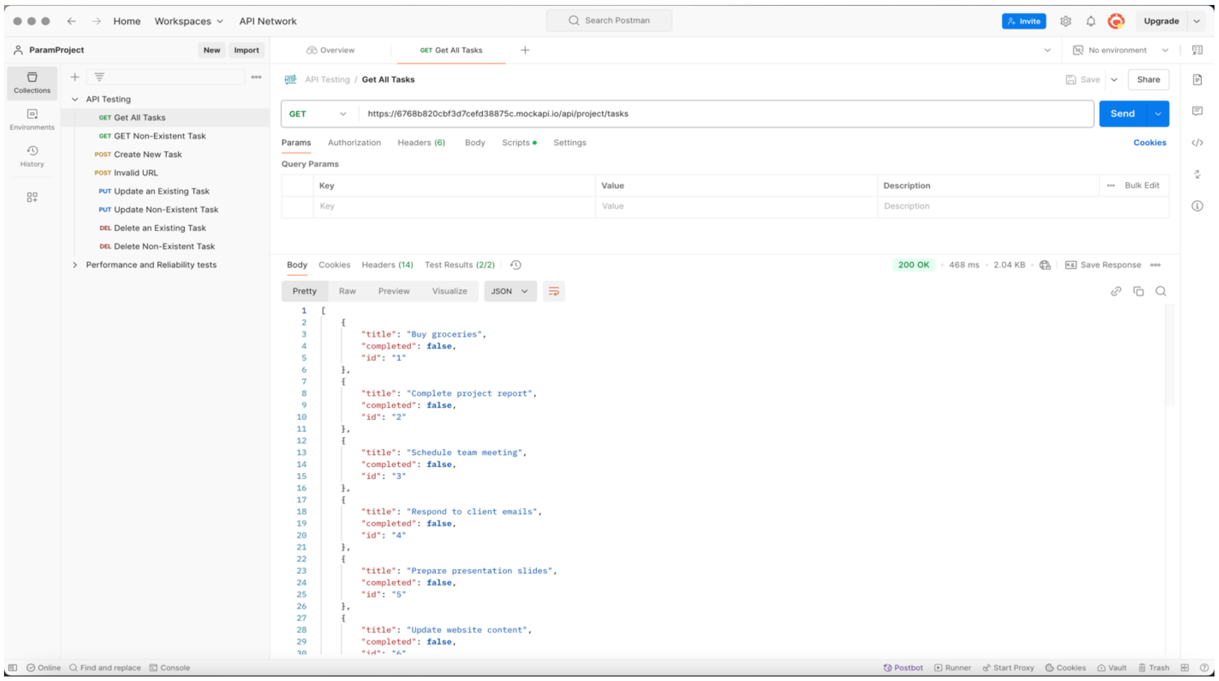
4.2. Performance Observations:

- Application remained responsive during high-frequency UI interactions.

5. Visuals

5.1. API Testing Screenshots

1. GET All Tasks (GET)

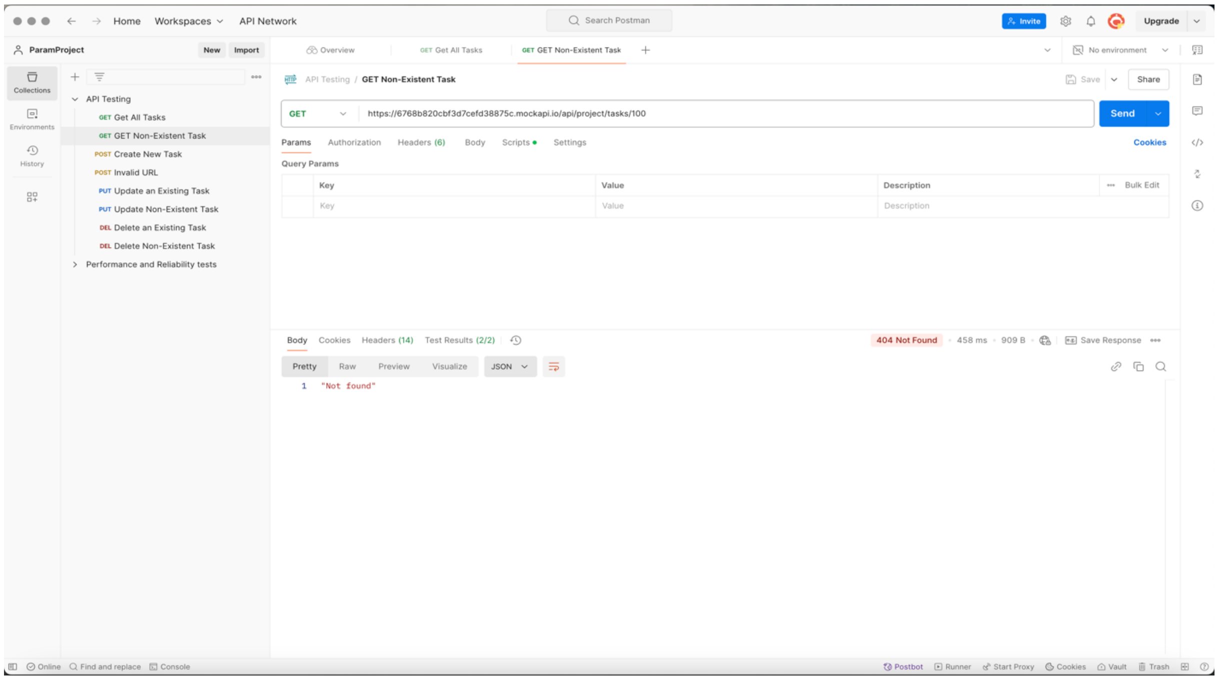


GET https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks

200 OK · 468 ms · 2.04 KB

```
[
    {
        "title": "Buy groceries",
        "completed": false,
        "id": "1"
    },
    {
        "title": "Complete project report",
        "completed": false,
        "id": "2"
    },
    {
        "title": "Schedule team meeting",
        "completed": false,
        "id": "3"
    },
    {
        "title": "Respond to client emails",
        "completed": false,
        "id": "4"
    },
    {
        "title": "Prepare presentation slides",
        "completed": false,
        "id": "5"
    },
    {
        "title": "Update website content",
        "completed": false,
        "id": "6"
```

2. GET Non-Existent Task (GET)

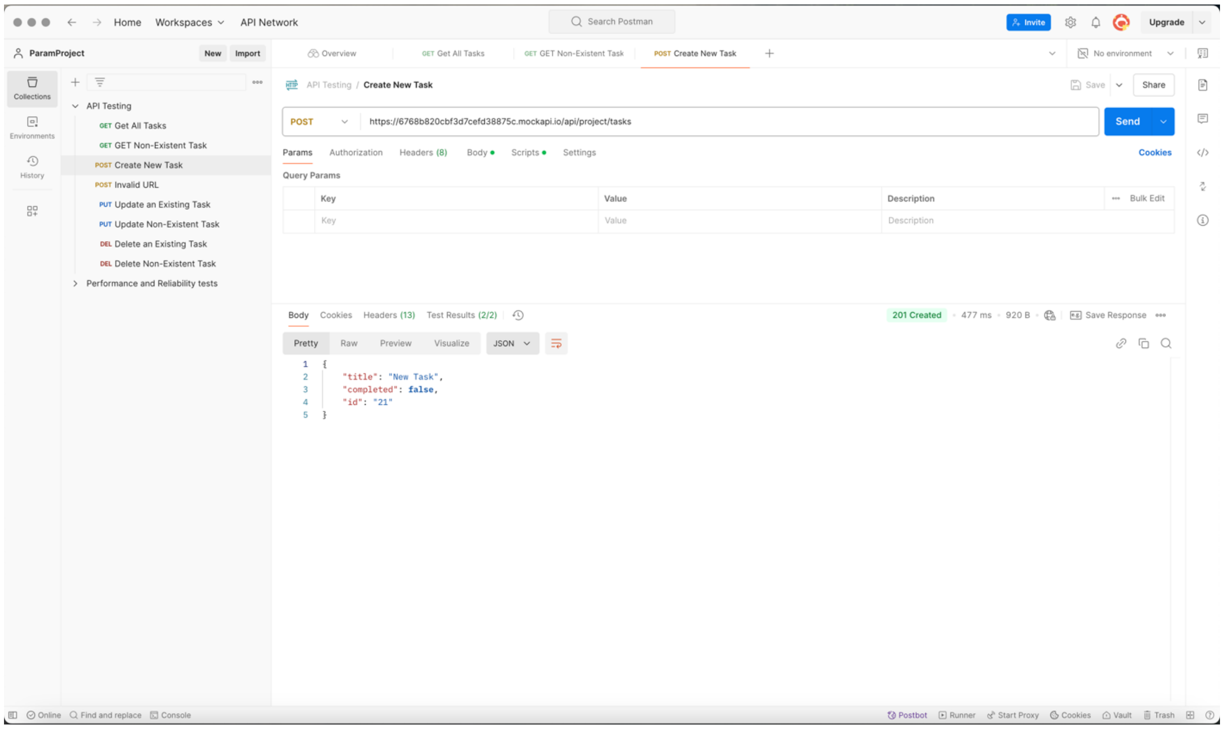


GET https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks/100

404 Not Found · 458 ms · 909 B

```
"Not found"
```

3. Create New Task (POST)

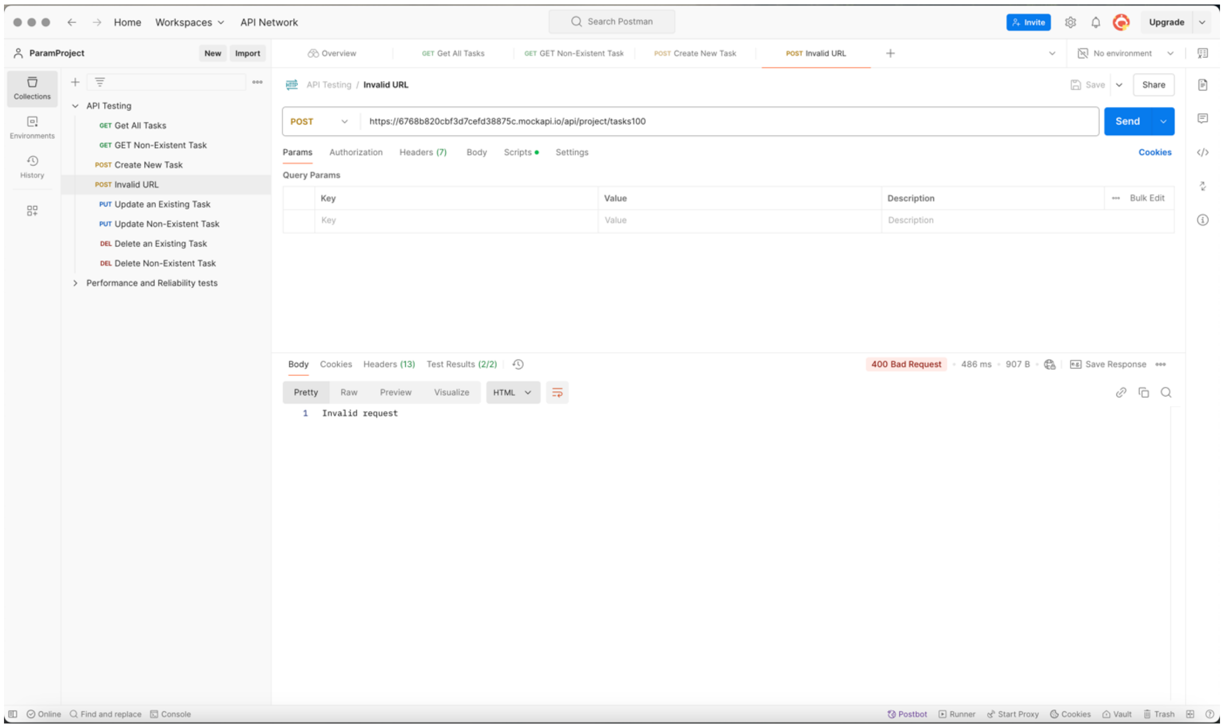


POST https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks

201 Created · 477 ms · 920 B

```
{
    "title": "New Task",
    "completed": false,
    "id": "21"
}
```

4. Invalid URL (POST):

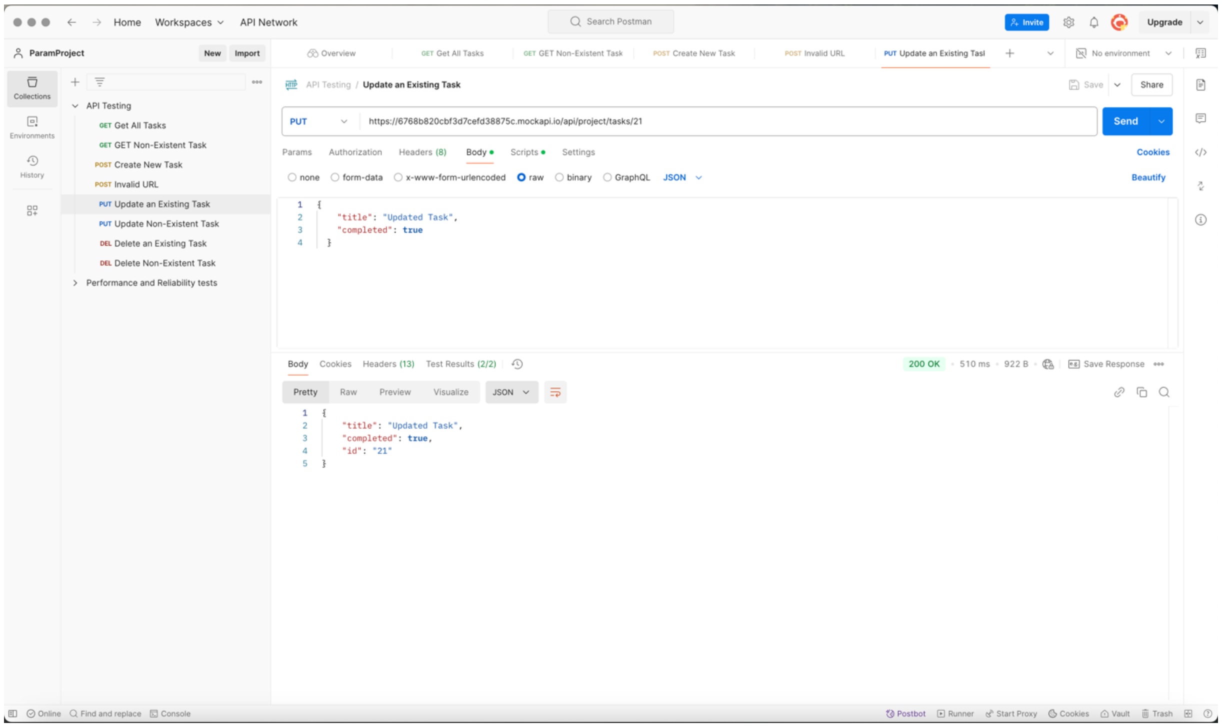


POST https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks100

400 Bad Request · 486 ms · 907 B

```
Invalid request
```

5. Update an Existing Task (PUT)



PUT https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks/21

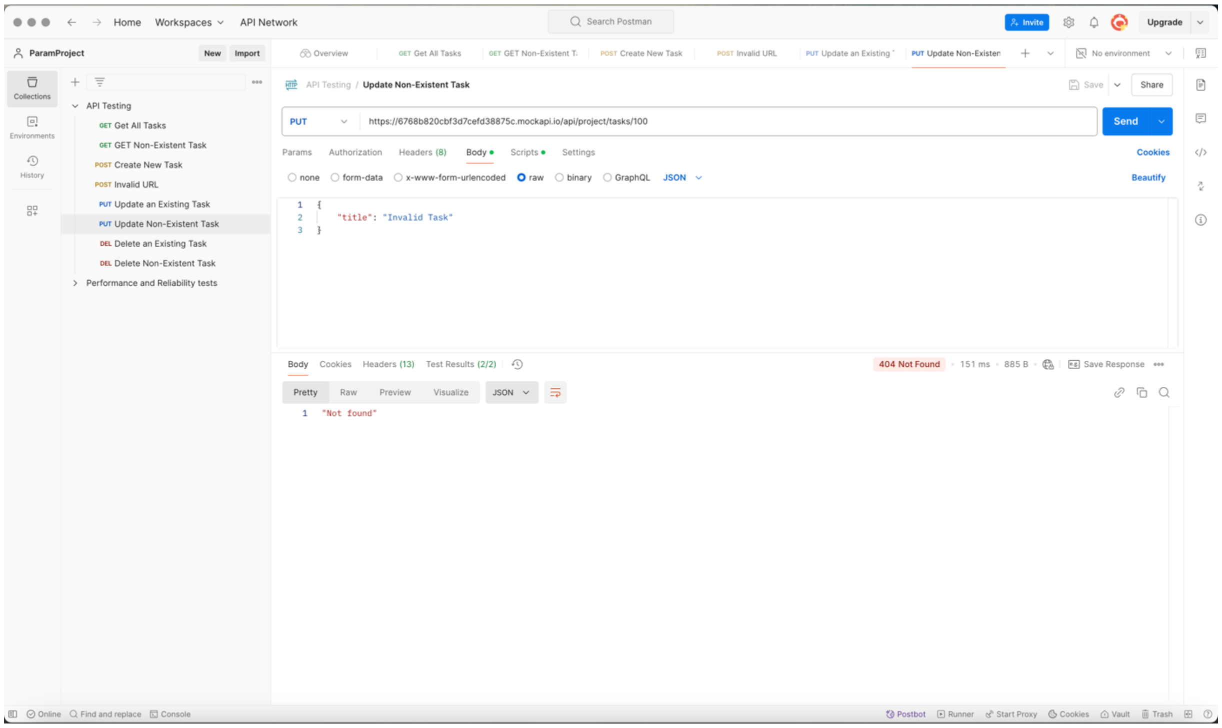
Body: raw JSON

```
{
    "title": "Updated Task",
    "completed": true
}
```

Response: 200 OK · 510 ms · 922 B

```
{
    "title": "Updated Task",
    "completed": true,
    "id": "21"
}
```

6. Update Non-Existent Task (PUT)



PUT https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks/100

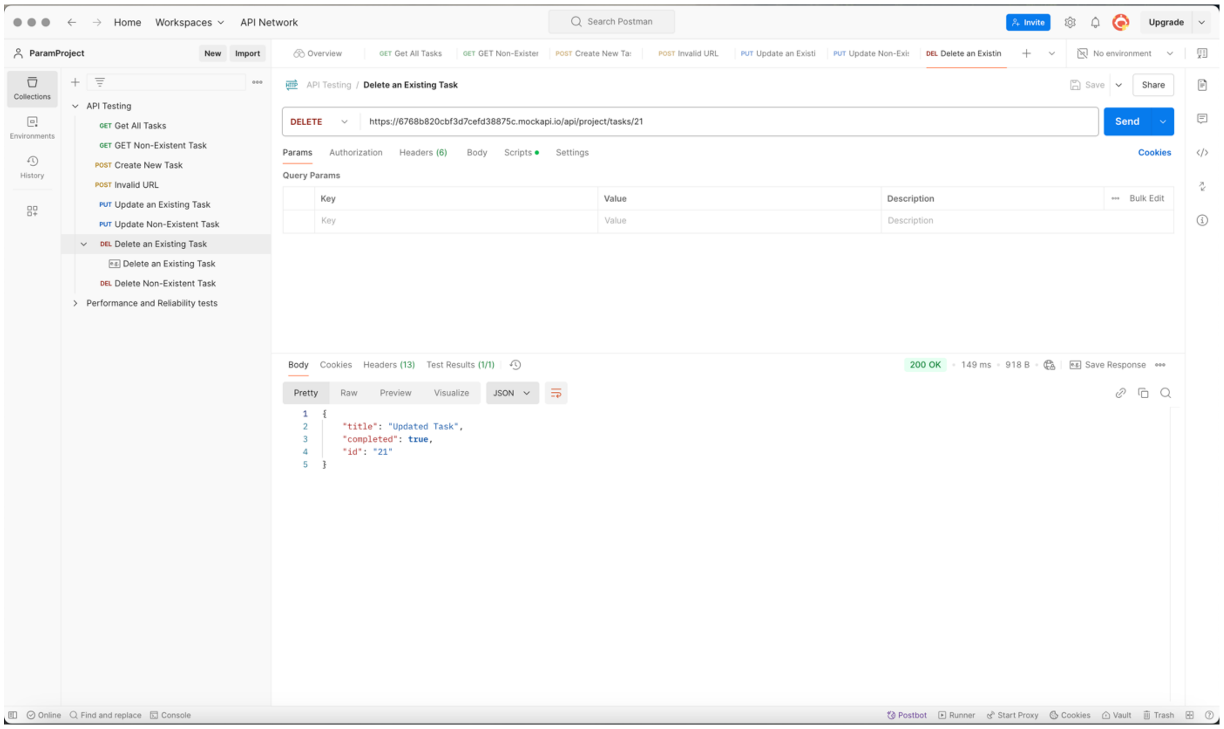
Body: raw JSON

```
{
    "title": "Invalid Task"
}
```

Response: 404 Not Found · 151 ms · 885 B

```
"Not found"
```

7. Delete an Existing Task (DELETE):

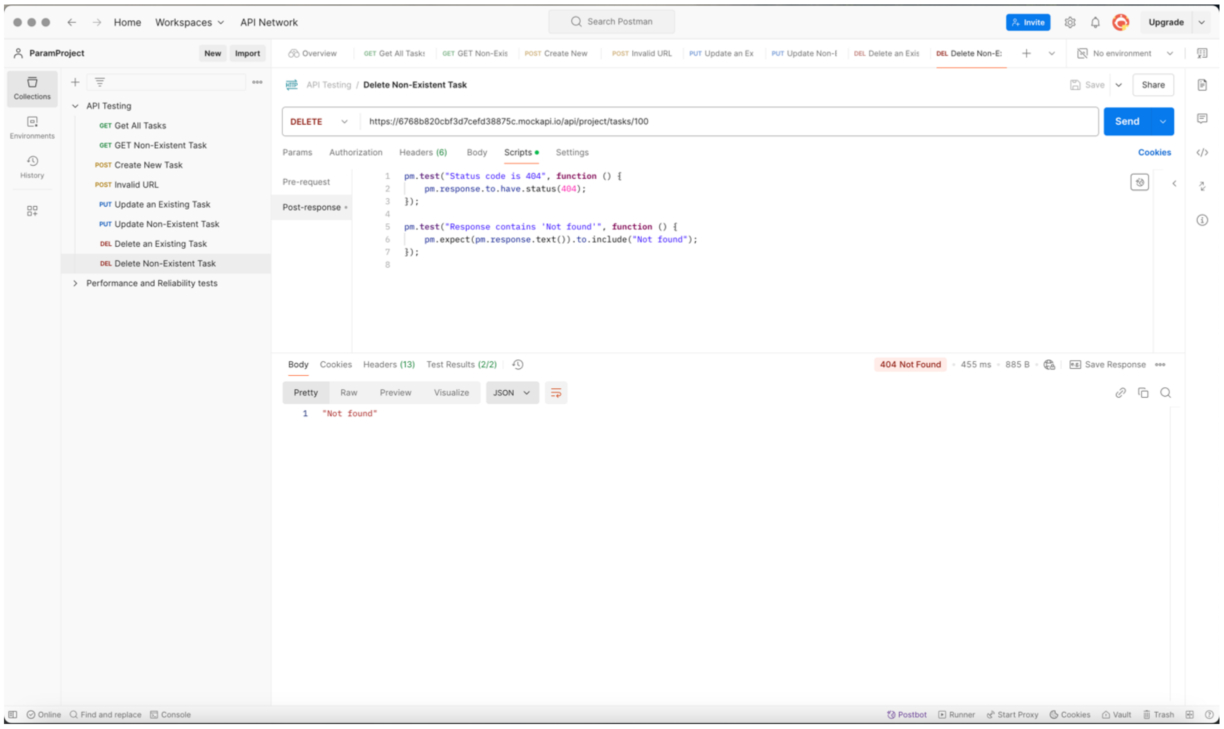


DELETE https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks/21

200 OK · 149 ms · 918 B

```
{
    "title": "Updated Task",
    "completed": true,
    "id": "21"
}
```

8. Delete Non-Existent Task (DELETE):



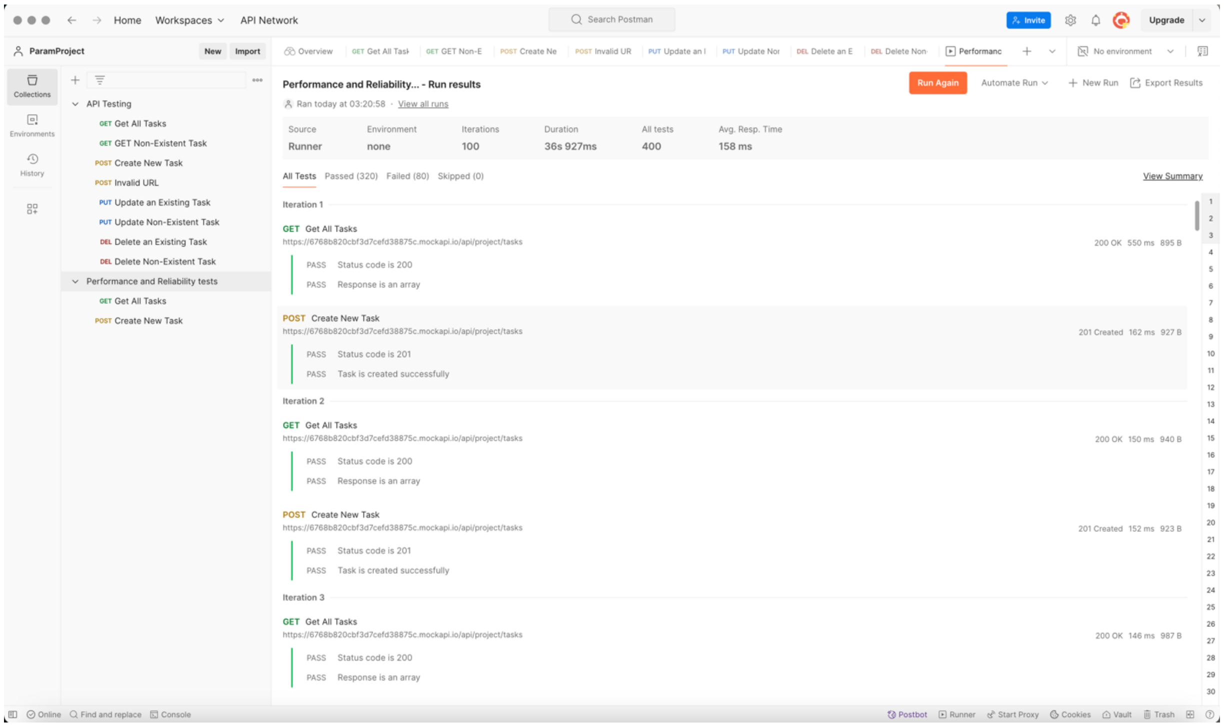
DELETE https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks/100

```
pm.test("Status code is 404", function () {
    pm.response.to.have.status(404);
});

pm.test("Response contains 'Not found'", function () {
    pm.expect(pm.response.text()).to.include("Not found");
});
```

404 Not Found · 455 ms · 885 B

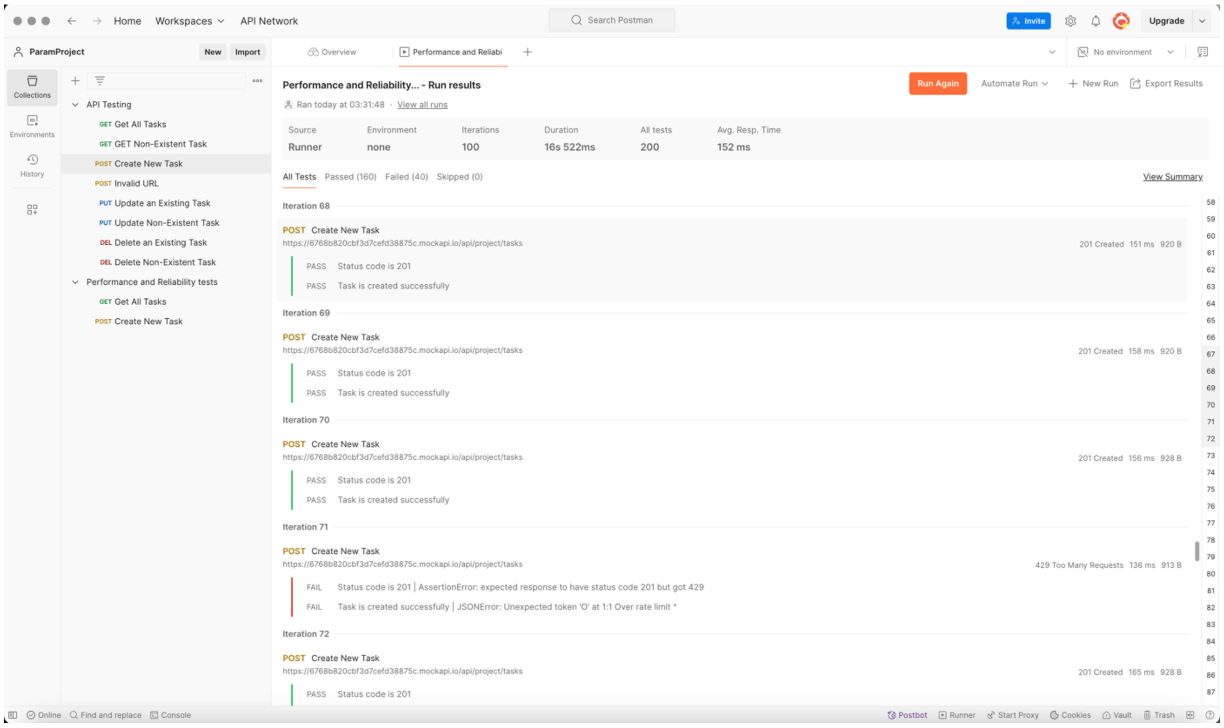
```
"Not found"
```

9. Simulated Load Test



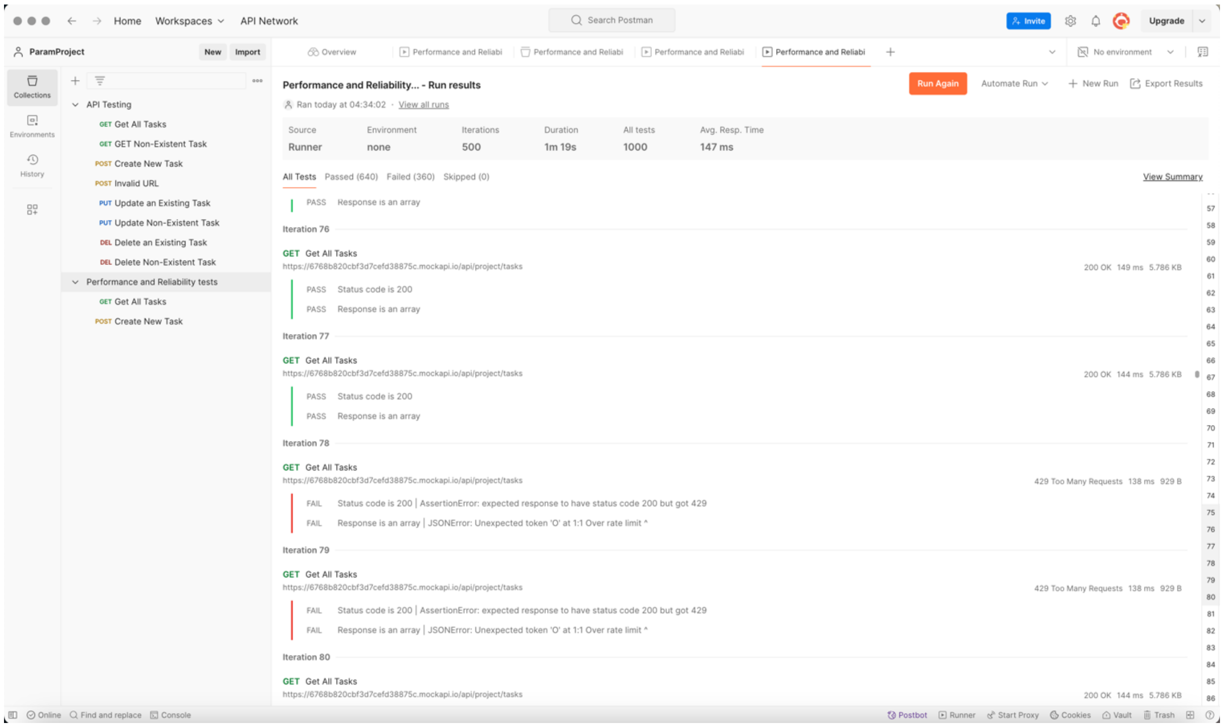
```
Performance and Reliability... - Run results
Ran today at 03:20:58 · View all runs

Source    Environment    Iterations    Duration     All tests    Avg. Resp. Time
Runner    none           100           36s 927ms    400          158 ms

All Tests   Passed (320)   Failed (80)   Skipped (0)

Iteration 1
GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          200 OK  550 ms  895 B
    PASS   Status code is 200
    PASS   Response is an array
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          201 Created  162 ms  927 B
    PASS   Status code is 201
    PASS   Task is created successfully
Iteration 2
GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          200 OK  150 ms  940 B
    PASS   Status code is 200
    PASS   Response is an array
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          201 Created  152 ms  923 B
    PASS   Status code is 201
    PASS   Task is created successfully
Iteration 3
GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          200 OK  146 ms  987 B
    PASS   Status code is 200
    PASS   Response is an array
```

10. Stress Test:



```
Performance and Reliability... - Run results
Ran today at 03:31:48 · View all runs

Source    Environment    Iterations    Duration     All tests    Avg. Resp. Time
Runner    none           100           16s 522ms    200          152 ms

All Tests   Passed (160)   Failed (40)   Skipped (0)

Iteration 68
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          201 Created  151 ms  920 B
    PASS   Status code is 201
    PASS   Task is created successfully
Iteration 69
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          201 Created  158 ms  920 B
    PASS   Status code is 201
    PASS   Task is created successfully
Iteration 70
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          201 Created  156 ms  928 B
    PASS   Status code is 201
    PASS   Task is created successfully
Iteration 71
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          429 Too Many Requests  136 ms  913 B
    FAIL   Status code is 201 | AssertionError: expected response to have status code 201 but got 429
    FAIL   Task is created successfully | JSONError: Unexpected token 'O' at 1:1 Over rate limit ^
Iteration 72
POST  Create New Task
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks          201 Created  165 ms  928 B
    PASS   Status code is 201
```

11. Reliability Test:



Performance and Reliability... - Run results

Ran today at 04:34:02 · View all runs

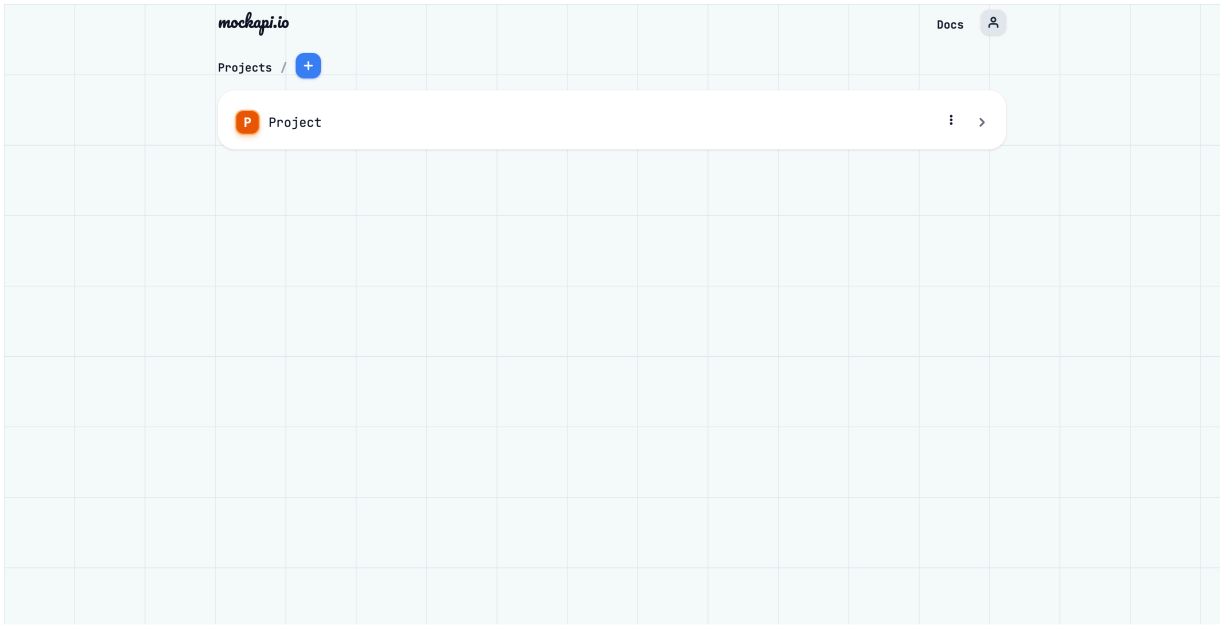
Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	none	500	1m 19s	1000	147 ms

All Tests   Passed (640)   Failed (360)   Skipped (0)

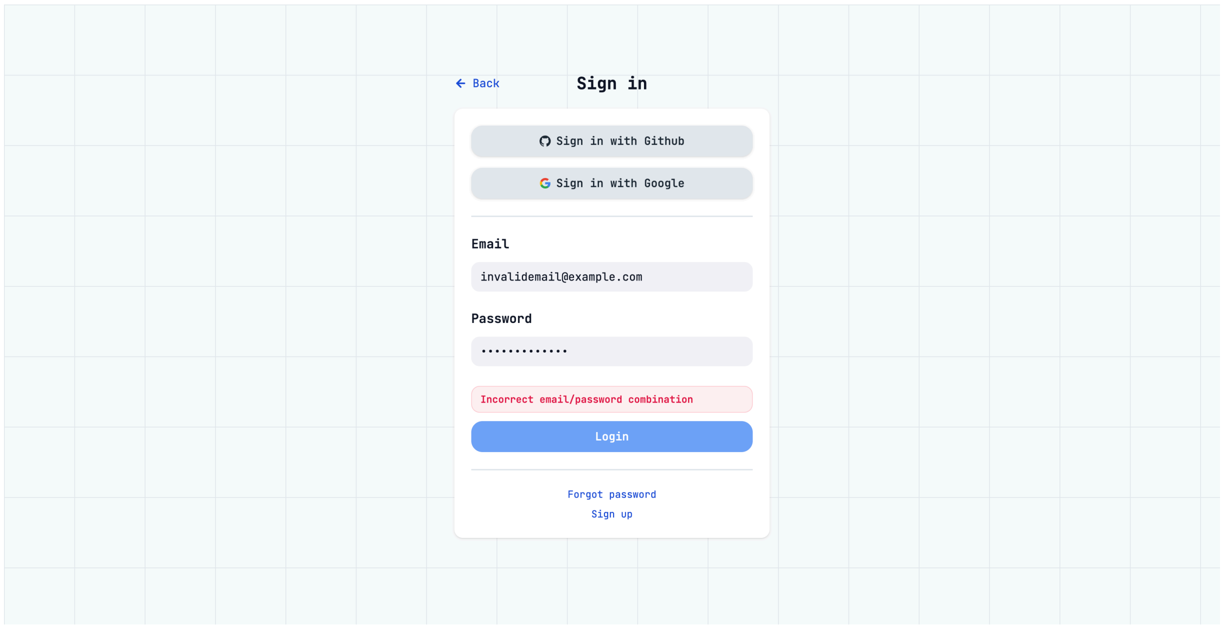
```
PASS   Response is an array

Iteration 76

GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks                200 OK  149 ms  5.786 KB
   PASS   Status code is 200
   PASS   Response is an array

Iteration 77

GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks                200 OK  144 ms  5.786 KB
   PASS   Status code is 200
   PASS   Response is an array

Iteration 78

GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks    429 Too Many Requests  138 ms  929 B
   FAIL   Status code is 200 | AssertionError: expected response to have status code 200 but got 429
   FAIL   Response is an array | JSONError: Unexpected token 'O' at 1:1 Over rate limit ^

Iteration 79

GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks    429 Too Many Requests  138 ms  929 B
   FAIL   Status code is 200 | AssertionError: expected response to have status code 200 but got 429
   FAIL   Response is an array | JSONError: Unexpected token 'O' at 1:1 Over rate limit ^

Iteration 80

GET  Get All Tasks
https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/tasks                200 OK  144 ms  5.786 KB
```

5.2. UI Testing Screenshots:

1. Valid Login:



2. Invalid Login:



Back

Sign in

Sign in with Github

Sign in with Google

Email

invalidemail@example.com

Password

•••••••••••••

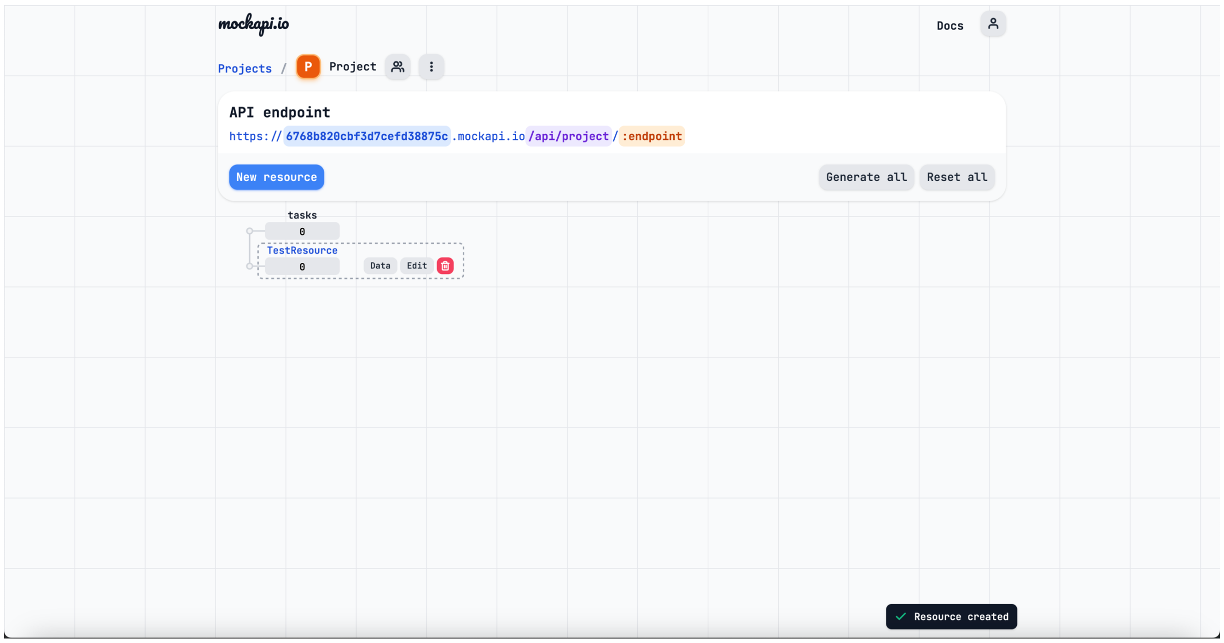
Incorrect email/password combination

Login

Forgot password

Sign up

3. Add Resource:



mockapi.io

Docs

Projects / P Project

API endpoint

https://6768b820cbf3d7cefd38875c.mockapi.io/api/project/:endpoint

New resource

Generate all

Reset all

tasks

0

TestResource

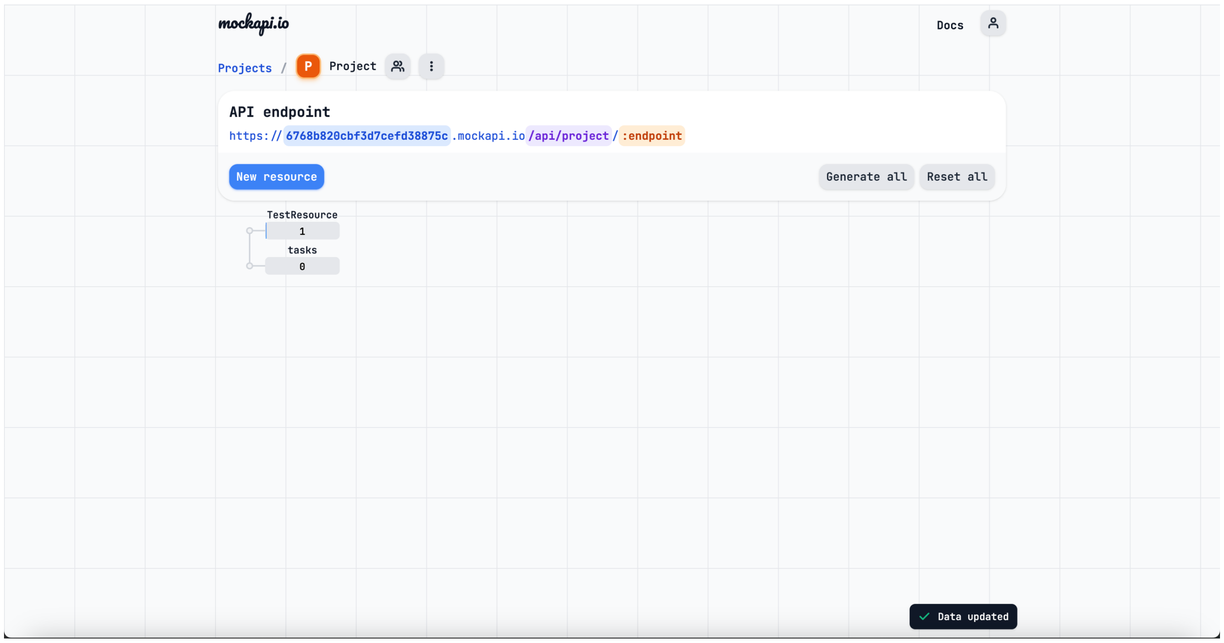
0

Data

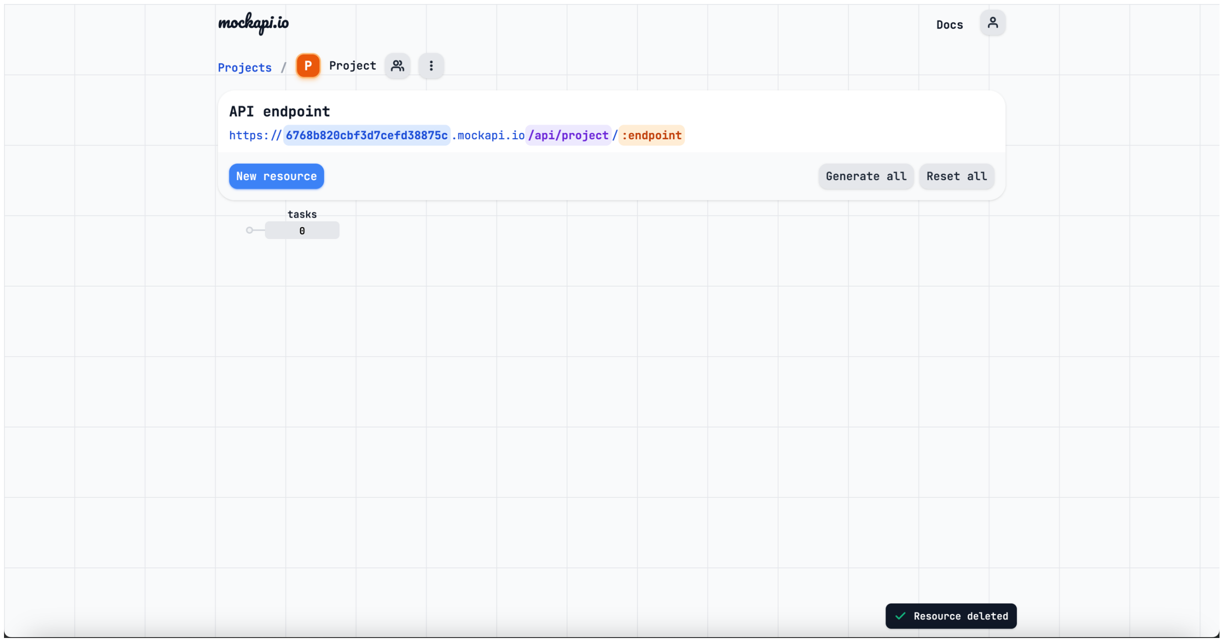
Edit

Resource created

4. **Edit Resource:**



5. **Delete Resource:**



6. Conclusion

In this project, we successfully tested the MockAPI application for both API and UI functionalities, meeting the objectives of the Software Quality Assurance Fundamentals course. API testing validated CRUD operations through positive and negative scenarios, with stress and reliability tests providing insights into the application’s performance under high load. UI testing automated workflows like login, resource creation, editing, and deletion, ensuring proper functionality and responsiveness. Despite challenges like API rate limits and UI stability, we resolved them effectively, gaining valuable experience with tools like Postman and Selenium. Overall, the project demonstrated a structured approach to software testing, highlighting the importance of quality assurance in ensuring application reliability and performance.

---

7. References and Tools

Tools Used:

1. **Postman:** API testing and performance evaluation.
2. **Selenium:** UI automation and interaction testing.
3. **Python:** Backend scripting for test automation.
4. **MockAPI:** Application under test.

Resources:

1. Official Selenium Documentation: https://www.selenium.dev/documentation/
2. Postman Learning Center: https://learning.postman.com/
3. MockAPI Documentation: https://github.com/mockapi-io/docs/wiki