POSTMAN

API testing with POSTMAN

By Földi Krizsán Ildikó

Why is API Functional testing important?

- identify bugs and errors early in the development process, before they cause problems in the final product saves time and money compared to fixing issues later on.

- have to validate if the API is functioning as expected and can handle different data types, requests, and responses
- check if the APIs are secure and data exchange between the application and the API is accurate and up to date

POSTMAN

Why Postman?

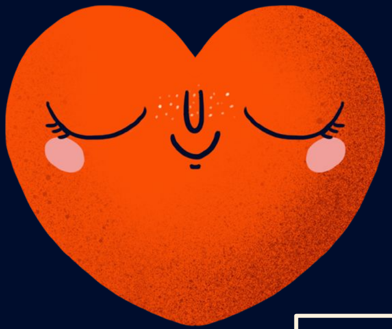
- Easy to use
- Supports a variety of request methods (GET, POST, PUT, PATCH, and DELETE)
- Collections and environments
- Test scripts

ATLASSIAN Developer

Why JIRA API?

- Jira is issue tracking and project management tool for software development teams
- through testing, we get to know the subject of the test in depth
- REST API has detailed documentation - but it is not easy to use

ATLASSIAN
Jira

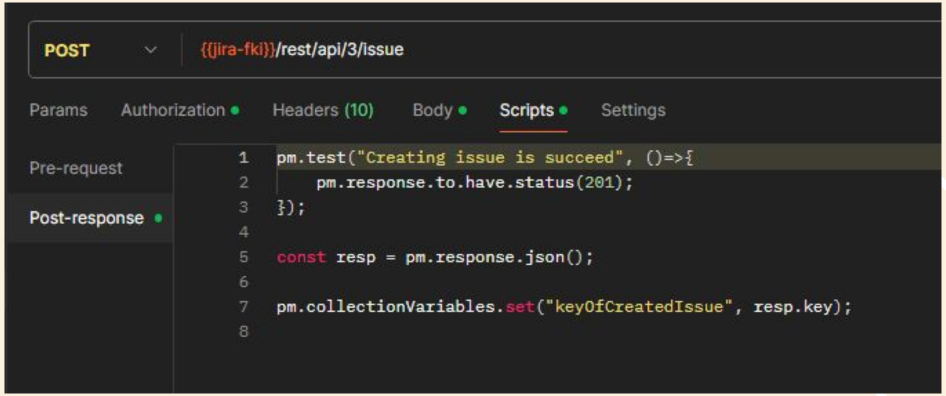


API Test Plan for Jira REST API

1. Get events/issues and validate that there are at least 1 event/issue
2. Create an issue via the Jira REST API and validate that the response code is 200
3. Get an issue by Key and validate the value of the summary field
4. Create a new comment to an existing Issue and validate it on the issue endpoint
5. Update an existing comment on an Issue and validate it on the issue endpoint
6. Delete an existing comment from an Issue and validate that it is not available on the issue endpoint anymore

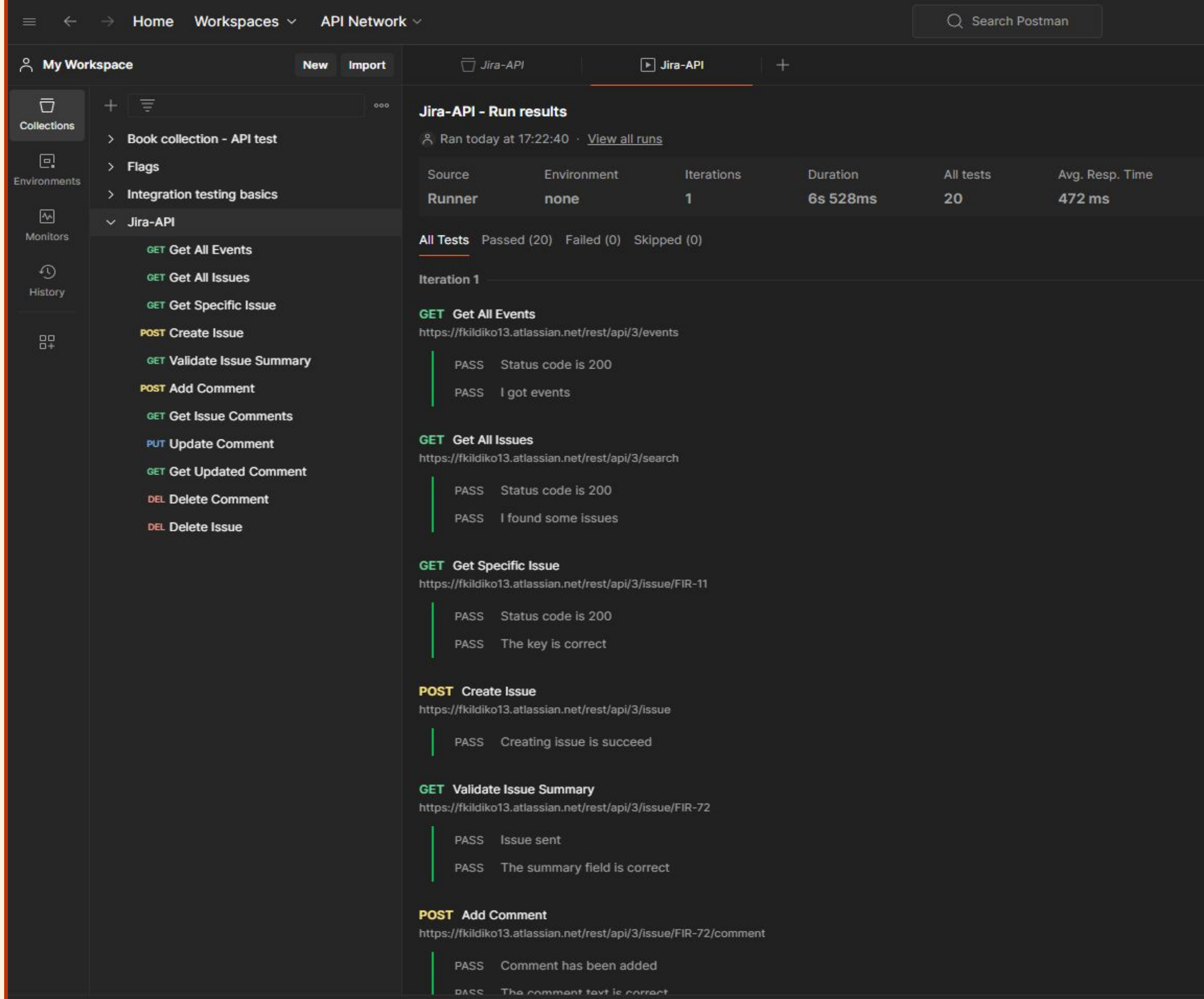
Create Collection

- Set up variables - base URL, data for authentication
- Create Test Cases, and run them manually
- Add tests for verification
- Store data from response
- Run collection
- Results can be exported as a JSON file



```
pm.test("Creating issue is succeed", ()=>{
    pm.response.to.have.status(201);
});

const resp = pm.response.json();

pm.collectionVariables.set("keyOfCreatedIssue", resp.key);
```

Results of running collection:



Book collection - API test
Flags
Integration testing basics
Jira-API
GET Get All Events
GET Get All Issues
GET Get Specific Issue
POST Create Issue
GET Validate Issue Summary
POST Add Comment
GET Get Issue Comments
PUT Update Comment
GET Get Updated Comment
DEL Delete Comment
DEL Delete Issue

Jira-API - Run results

Ran today at 17:22:40 · View all runs

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	none	1	6s 528ms	20	472 ms

All Tests Passed (20) Failed (0) Skipped (0)

Iteration 1

GET Get All Events
https://fkildiko13.atlassian.net/rest/api/3/events
PASS Status code is 200
PASS I got events

GET Get All Issues
https://fkildiko13.atlassian.net/rest/api/3/search
PASS Status code is 200
PASS I found some issues

GET Get Specific Issue
https://fkildiko13.atlassian.net/rest/api/3/issue/FIR-11
PASS Status code is 200
PASS The key is correct

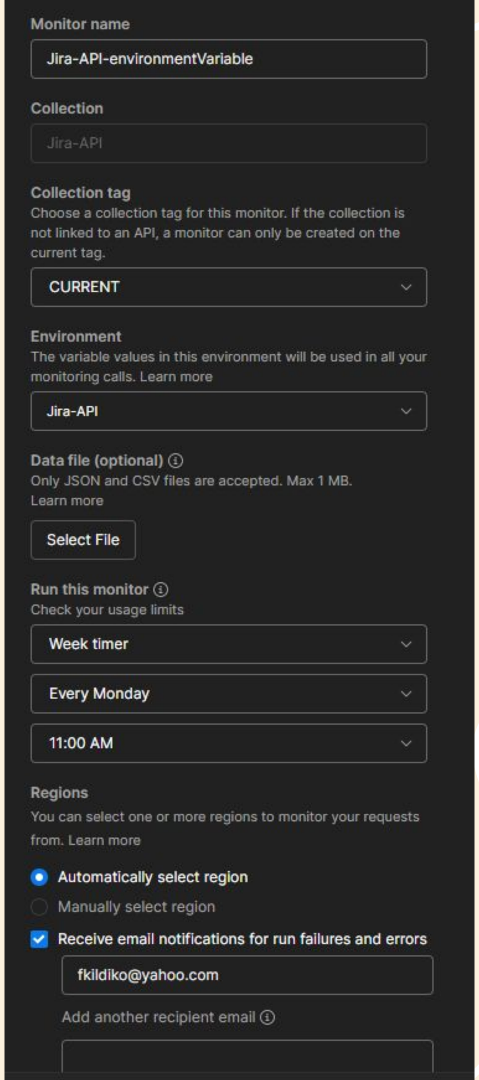
POST Create Issue
https://fkildiko13.atlassian.net/rest/api/3/issue
PASS Creating issue is succeed

GET Validate Issue Summary
https://fkildiko13.atlassian.net/rest/api/3/issue/FIR-72
PASS Issue sent
PASS The summary field is correct

POST Add Comment
https://fkildiko13.atlassian.net/rest/api/3/issue/FIR-72/comment
PASS Comment has been added
PASS The comment text is correct

Monitoring:

- continuously check the health of your APIs
- Set up the environment for variables
- set the time interval at which you want to run the collection
- Set up an email to get a notification if tests failed



Monitor name
Jira-API-environmentVariable

Collection
Jira-API

Collection tag
Choose a collection tag for this monitor. If the collection is not linked to an API, a monitor can only be created on the current tag.
CURRENT

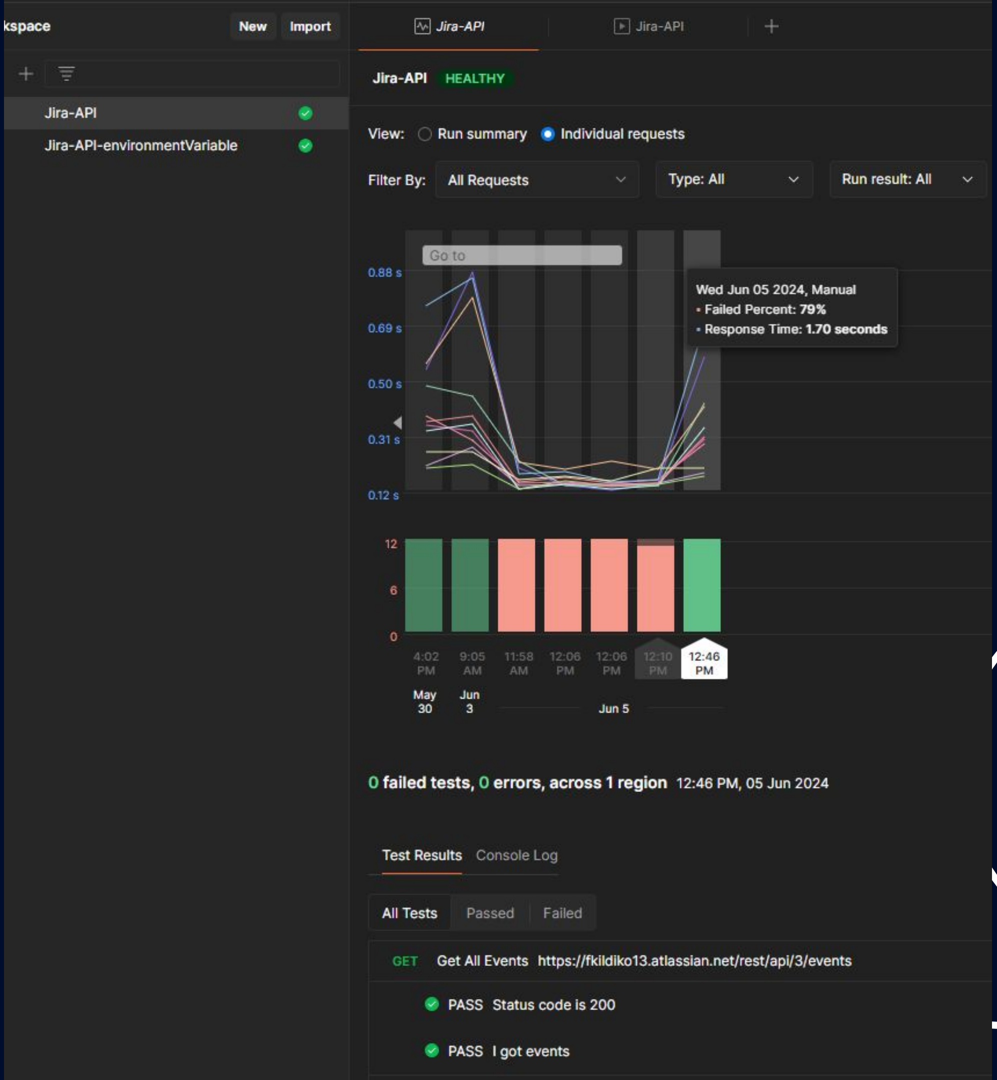
Environment
The variable values in this environment will be used in all your monitoring calls. Learn more
Jira-API

Data file (optional)
Only JSON and CSV files are accepted. Max 1 MB. Learn more
Select File

Run this monitor
Check your usage limits
Week timer
Every Monday
11:00 AM

Regions
You can select one or more regions to monitor your requests from. Learn more
[✓ Automatically select region]
Manually select region
[✓] Receive email notifications for run failures and errors
fkildiko@yahoo.com
Add another recipient email

Report of monitoring:



Jira-API HEALTHY

View: ○ Run summary [✓ Individual requests]

Filter By: All Requests | Type: All | Run result: All

Wed Jun 05 2024, Manual
- Failed Percent: 79%
- Response Time: 1.70 seconds

0 failed tests, 0 errors, across 1 region 12:46 PM, 05 Jun 2024

GET Get All Events https://fkildiko13.atlassian.net/rest/api/3/events

✓ PASS Status code is 200

✓ PASS I got events

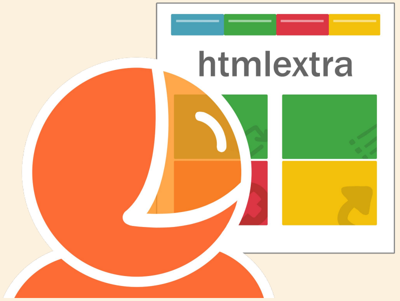
Using GitHub Actions for Automated Workflows

- Automatically running tests whenever you push changes to main branch
- Export your Postman collection as a JSON file.
- If you use environment variables, export them as a separate JSON file as well.
- Install Newman for running tests from terminal
- Unfortunately, secret variables are visible within the JSON file, posing a security risk.

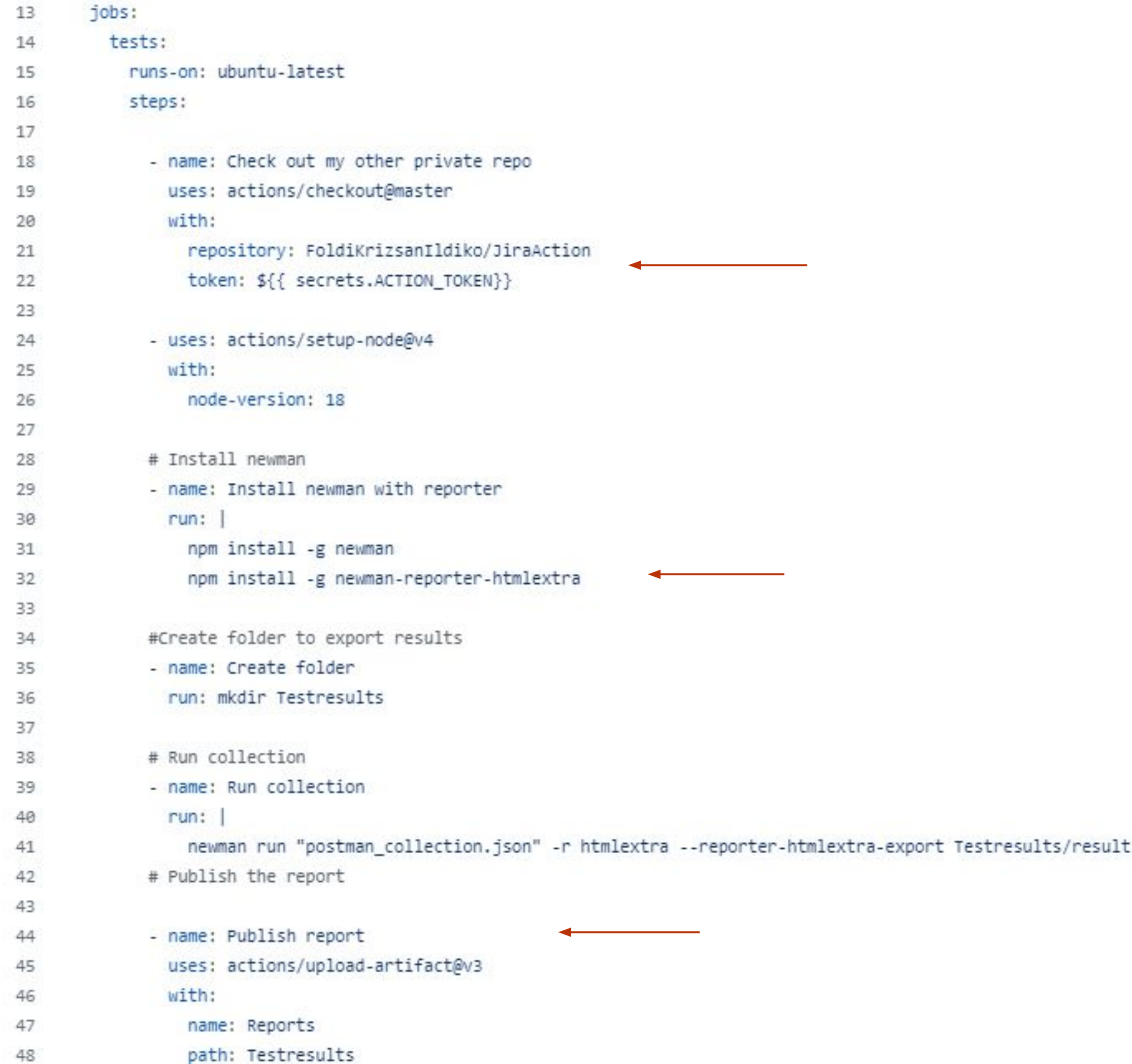
My solution for keeping data safe:

- I created a separate private repository to store the collection file containing sensitive information.
- In pipeline of my code repository I can access with token the private repository, and run all tests from there
- Saves me setting up secrets in repository and modifying the collection file to reference those variables
- Yaml file is more simple

Jobs of workflow file:

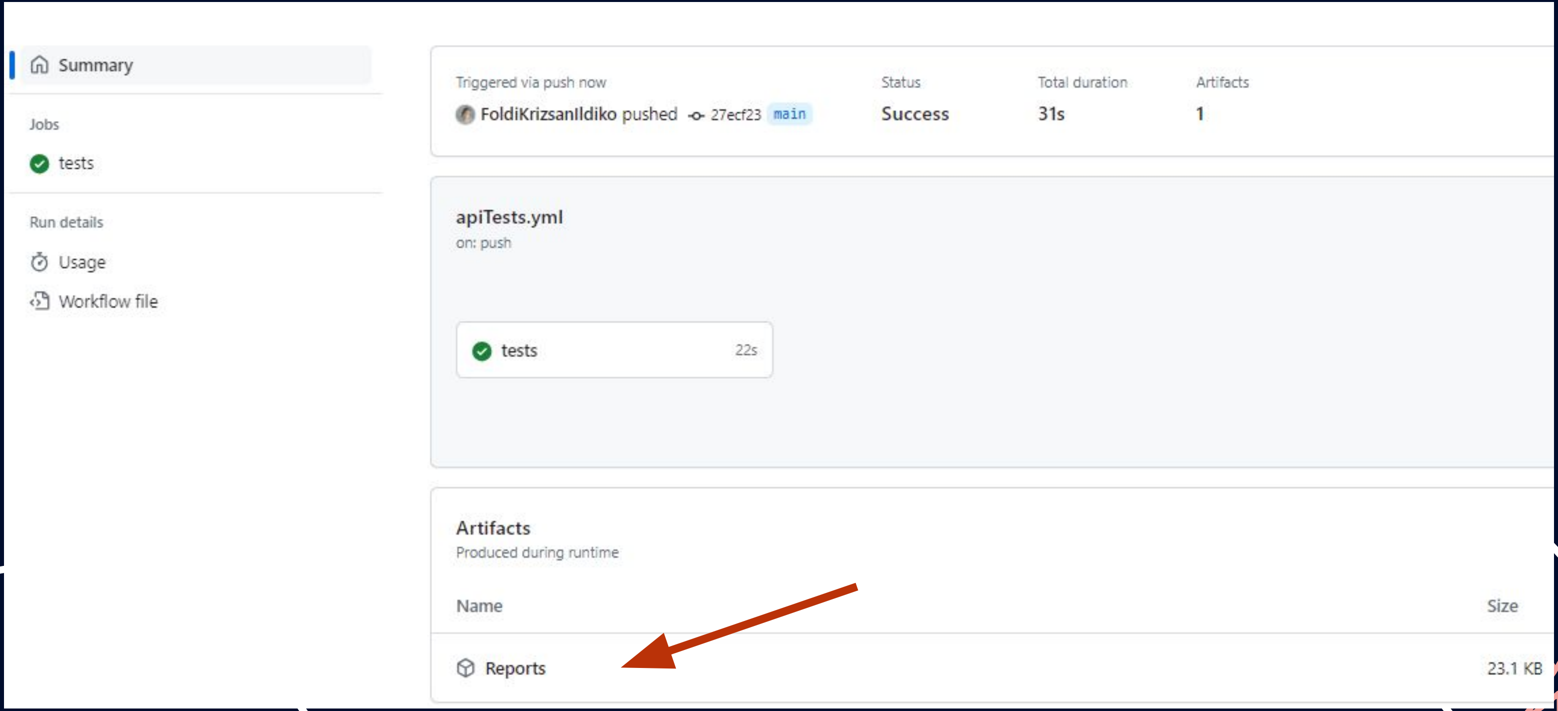


We can generate HTML reports for the tests and store them in a dedicated folder.



```yaml
jobs:
  tests:
    runs-on: ubuntu-latest
    steps:

      - name: Check out my other private repo
        uses: actions/checkout@master
        with:
          repository: FoldiKrizsanIldiko/JiraAction
          token: ${{ secrets.ACTION_TOKEN}}

      - uses: actions/setup-node@v4
        with:
          node-version: 18

      # Install newman
      - name: Install newman with reporter
        run: |
          npm install -g newman
          npm install -g newman-reporter-htmlextra

      #Create folder to export results
      - name: Create folder
        run: mkdir Testresults

      # Run collection
      - name: Run collection
        run: |
          newman run "postman_collection.json" -r htmlextra --reporter-htmlextra-export Testresults/result
      # Publish the report

      - name: Publish report
        uses: actions/upload-artifact@v3
        with:
          name: Reports
          path: Testresults
```

GitHub Actions workflow:



Summary

Jobs

tests

Run details

Usage

Workflow file

Triggered via push now

FoldiKrizsanIldiko pushed 27ecf23 main

Status

Success

Total duration

31s

Artifacts

1

apiTests.yml

on: push

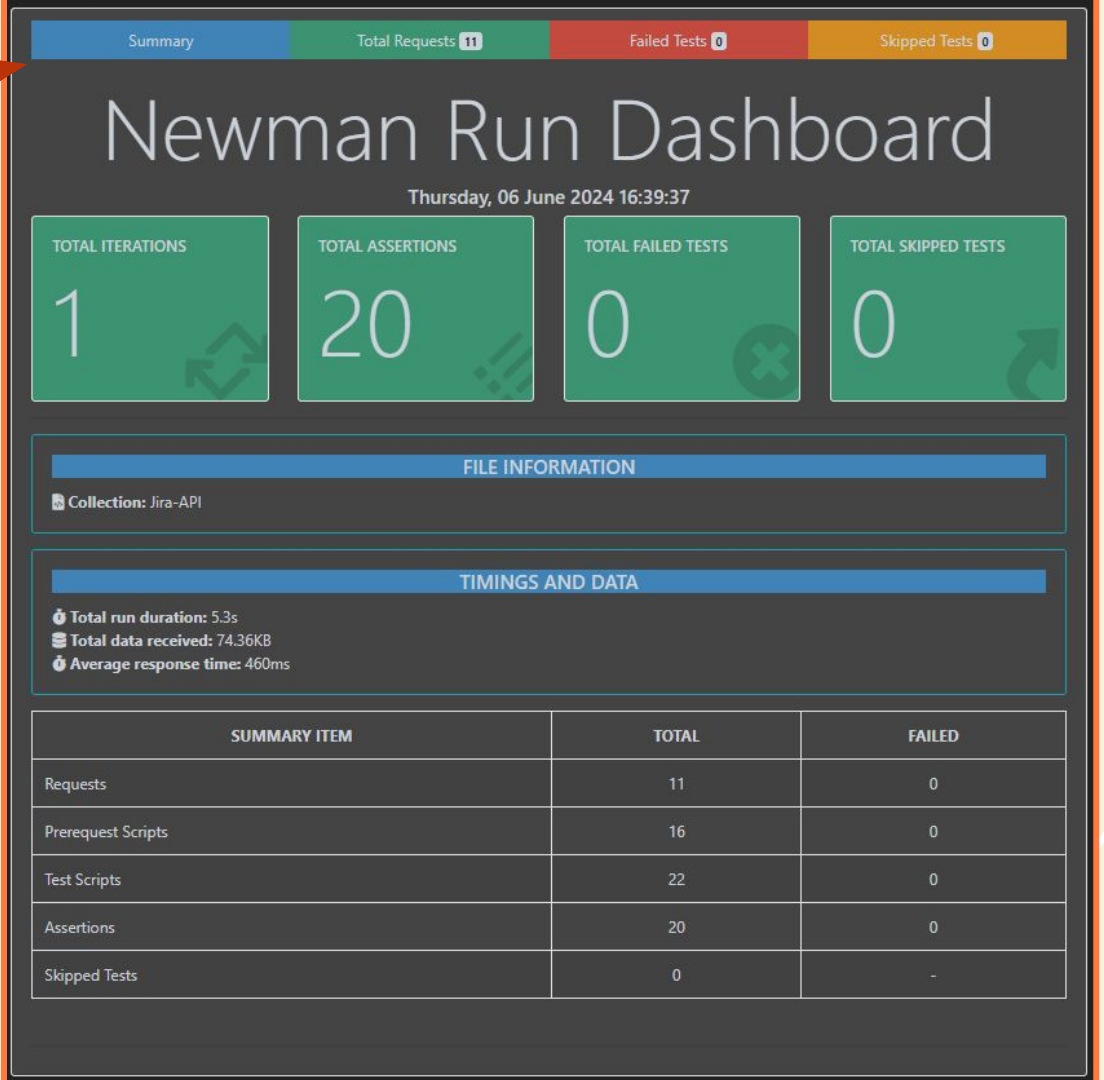
tests 22s

Artifacts

Produced during runtime

Name	Size
Reports	23.1 KB

HTML report summary:



Summary | Total Requests 11 | Failed Tests 0 | Skipped Tests 0

Newman Run Dashboard

Thursday, 06 June 2024 16:39:37

TOTAL ITERATIONS	TOTAL ASSERTIONS	TOTAL FAILED TESTS	TOTAL SKIPPED TESTS
1	20	0	0

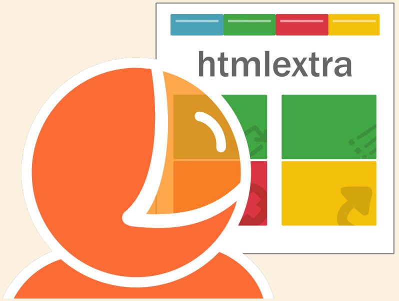
FILE INFORMATION

Collection: Jira-API

TIMINGS AND DATA

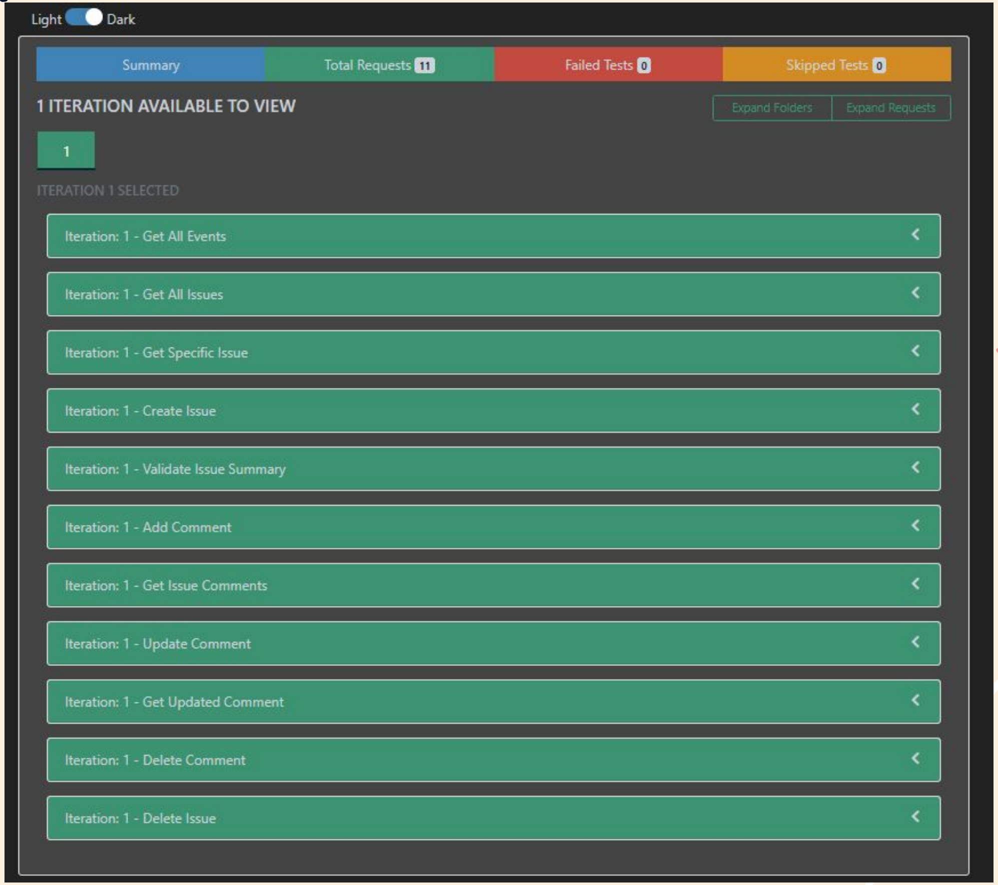
Total run duration: 5.3s
Total data received: 74.36KB
Average response time: 460ms

SUMMARY ITEM	TOTAL	FAILED
Requests	11	0
Prerequest Scripts	16	0
Test Scripts	22	0
Assertions	20	0
Skipped Tests	0	-



HTML Report Requests:

- All the API requests are expandable, providing access to complete details including requests, responses, and corresponding tests.



Light Dark

Summary | Total Requests 11 | Failed Tests 0 | Skipped Tests 0

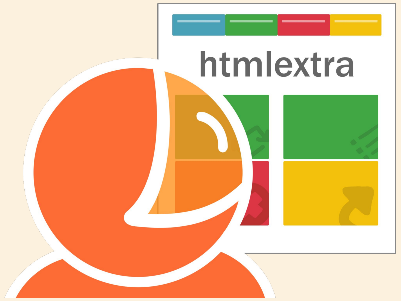
1 ITERATION AVAILABLE TO VIEW

Expand Folders | Expand Requests

1

ITERATION 1 SELECTED

- Iteration: 1 - Get All Events
- Iteration: 1 - Get All Issues
- Iteration: 1 - Get Specific Issue
- Iteration: 1 - Create Issue
- Iteration: 1 - Validate Issue Summary
- Iteration: 1 - Add Comment
- Iteration: 1 - Get Issue Comments
- Iteration: 1 - Update Comment
- Iteration: 1 - Get Updated Comment
- Iteration: 1 - Delete Comment
- Iteration: 1 - Delete Issue



Thanks!

Do you have **any questions?**