Devops Internal Assessment 1 Report on

“Katalon Studio”

by

16010122243: **Ayush Nayak**
C-2

For the subject

Devops

Department of Computer Engineering
K. J. Somaiya College of Engineering
(Constituent College of Somaiya Vidyavihar University)
Academic Year 2025-26

Introduction to Katalon Studio

Katalon Studio stands out as a powerful and versatile open-source test automation framework. Its primary function is to streamline automated testing for web, mobile, desktop, and API applications, ensuring high-quality software delivery across diverse platforms.

Katalon Studio is widely used by QA engineers, software testers, and developers to simulate real-world user interactions and validate application behavior. It has become a staple in the toolkit of quality assurance teams and DevOps professionals, offering a reliable way to automate regression tests, functional tests, integration tests, and performance checks within controlled environments. By replicating real-world user scenarios, teams can effectively assess application reliability and uncover weaknesses before software reaches end-users.

The significance of Katalon Studio lies in its ability to simplify and accelerate the testing lifecycle. While manual testing is often time-consuming, repetitive, and error-prone, Katalon Studio automates the process, increasing test coverage and reducing human effort. Its intuitive interface and scriptless test creation features make it accessible for beginners, while its advanced scripting capabilities with Groovy and Java cater to experienced automation engineers.

Katalon Studio also addresses the growing demand for **CI/CD integration** in modern software development. It integrates seamlessly with tools such as Jenkins, Git, Azure DevOps, and Docker, enabling teams to embed testing into continuous delivery pipelines. This ensures that defects are detected early in the development cycle, reducing risks, improving speed, and enhancing overall product quality.

The toolkit provides a platform to emulate various testing scenarios, offering a comprehensive approach to software quality assurance. With features like cross-browser testing, API request chaining, data-driven testing, reporting dashboards, and analytics, Katalon Studio not only improves efficiency but also raises awareness within teams about the importance of consistent and reliable testing practices.

Features and Characteristics of Katalon Studio

Accessibility and Platform Integration:
Katalon Studio runs on Windows, macOS, and Linux and provides installers and standalone executables for common environments. It offers integrations and plugins for popular development and collaboration tools, which makes it easy to include automated testing in existing toolchains and development workflows.

User-Friendly Interface:
Katalon Studio provides an intuitive, GUI-driven IDE with visual test case designers, record-and-playback tools, and an object repository. This lowers the barrier to entry for testers who are new to automation while retaining advanced options for power users. The visual test flows and built-in templates simplify building and maintaining test suites.

Versatile Testing Vectors:
Katalon supports a wide spectrum of testing types so teams can centralize automation across projects:

- **Web testing:** Cross-browser automation for modern browsers and headless modes.

- **Mobile testing:** Native, hybrid, and web app testing on Android and iOS (real devices and emulators).

- **API testing:** REST and SOAP request creation, chaining, validation, and data-driven API scenarios.

- **Desktop testing:** Support for desktop application automation (where applicable through plugins/integrations).

- **Integration testing:** Facilities to combine UI and API checks in end-to-end scenarios.

Specific Testing Capabilities:

Katalon Studio includes many concrete capabilities that make it practical and powerful for real-world QA work:

- **Record & Playback and Object Spy:** Quickly capture UI interactions and object locators, then refine them in the object repository.

- **Scriptless & Scripted Modes:** Non-technical testers can use keyword-driven or visual test creation, while advanced users can write and extend tests in Groovy/Java for finer control.

- **Data-Driven Testing:** Drive tests with CSV, Excel, databases, or other data sources to run many scenarios from the same script.

- **Cross-Browser Execution & Parallelization:** Execute tests across multiple browsers and run tests in parallel to shorten feedback cycles.

- **CI/CD Integration:** Native or plugin-based integrations with Jenkins, Git, Azure DevOps, GitHub Actions, and Docker so tests can run automatically in pipelines.

- **Reporting & Analytics:** Built-in execution reports and logs, plus dashboards and test metrics to track test health, pass/fail trends, and flakiness.

- **Test Orchestration & Management:** Integrations with test management tools and a cloud-based option (TestOps/TestCloud) for scheduling, environment control, and centralized result storage.

- **Reusable Test Components:** Support for modular test design—keywords, custom keywords, and shared libraries—encouraging maintainable test suites.

<u>Details of the functionalities implemented:</u>

The list of features/functionalities proposed to be implemented are:

1) Basic Web Smoke Test (Demo test case)

What was implemented

- A simple smoke test that opens `https://example.com`, verifies visible content, extracts text from heading and paragraph, and takes a screenshot.

Detailed functionality

- Open browser (Chrome/Firefox).
- Navigate to `https://example.com`.
- Verify the heading text `Example Domain` is present.
- Verify paragraph (`<p>`) is present and log its text to the test log.
- Save a screenshot (`Reports/example_page.png`).
- Close the browser.

How it works (technical)

- Implemented as a Katalon Test Case `TC_OpenExample` using Groovy + Katalon WebUI keywords (`openBrowser`, `navigateToUrl`, `verifyTextPresent`, `getText`, `takeScreenshot`, `closeBrowser`).
- Uses dynamic `TestObject` instances created in script (XPath `//h1` and `//p`) so we avoid over-reliance on the Object Repository for this simple demo.

2) Test Suite & Test Suite Collection

What we implemented

- A Test Suite `TS_Demo` that contains the `TC_OpenExample` test case to run grouped tests.

Detailed functionality

- Grouping of test cases so CI can trigger a single suite instead of individual tests.
- Easy to extend later with more tests.

How it works

- Katalon Test Suite file references included test cases and execution profile.

3) Screenshot and Report Generation

What we implemented

- Screenshots saved during the run and standard Katalon Reports (HTML / JUnit / CSV).

Detailed functionality

- `WebUI.takeScreenshot('Reports/example_page.png')` stores a snapshot.
- Katalon automatically generates run logs and a report folder under `Reports/`.

Files changed / produced

- `Reports/example_page.png` (artifact produced at runtime).

- `Reports/<timestamp>/` containing HTML/JUnit/CSV logs (runtime output, typically not committed).

What you'll see on GitHub / CI

- On GitHub: we **do not** commit runtime `Reports/` by default (they're artifacts).
- On CI (GitHub Actions): the workflow uploads `Reports/` as artifacts — available to download from the Actions run.

4) Git Integration (project version control)

What we implemented

- Connected the Katalon project to your GitHub repo, committed initial project files, and pushed.

Detailed functionality

- Local Git repo created (or initialized by Katalon's Team → Share Project).
- First commit and push to `main` branch.
- Ongoing workflow: use Katalon Studio's Team → Commit / Push / Pull for daily changes.

Files changed

- `.gitignore` (recommended) to exclude `bin/`, temp files, and sometimes `Reports/`.
- All project metadata files under repo tracked in Git.

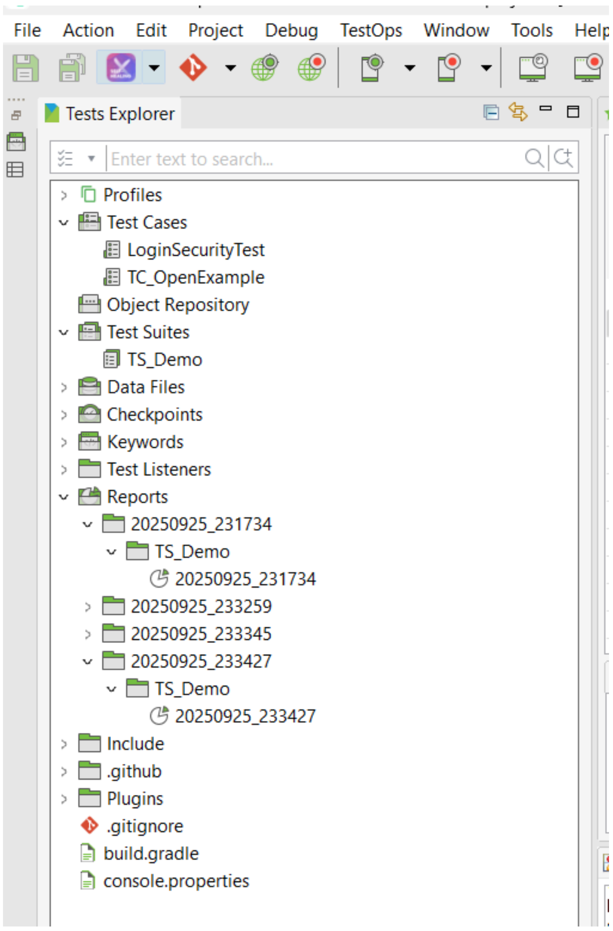
Results & Outcomes:

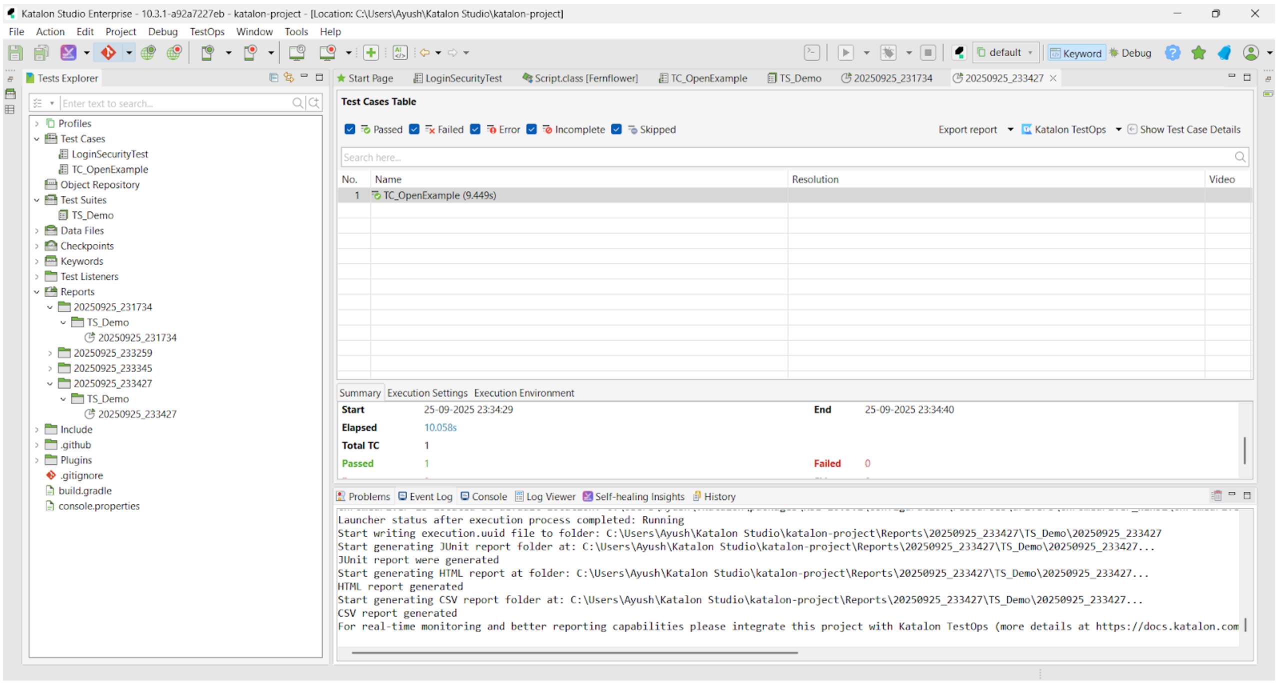
The problem addressed by the Katalon Studio implementation is the need for automated, repeatable, and reliable testing of web applications. Manual testing is often time-consuming, prone to human error, and inconsistent, which can lead to undetected issues in production. By automating simple scenarios like verifying page content, capturing screenshots, and logging results, the project demonstrates how repetitive validation tasks can be streamlined to save time and improve accuracy in software quality assurance.

The study aims to tackle the challenge of ensuring web applications function as expected across different environments. Traditional manual methods may overlook small UI or content changes, while automation provides consistent and objective checks. By running these tests both locally and through GitHub Actions, the project showcases how developers can integrate automated validation into their workflow, ensuring that even minor changes are immediately tested and verified.

The expected outcome of this project is to provide a controlled environment where basic automated tests run reliably and generate reports that can be reviewed by teams. This enables faster feedback on application stability, improved collaboration through shared results in GitHub, and a foundation for scaling test coverage. Ultimately, the Katalon implementation helps improve efficiency, reduce risk of unnoticed bugs, and strengthens overall software quality assurance practices.

Results(Screenshots):





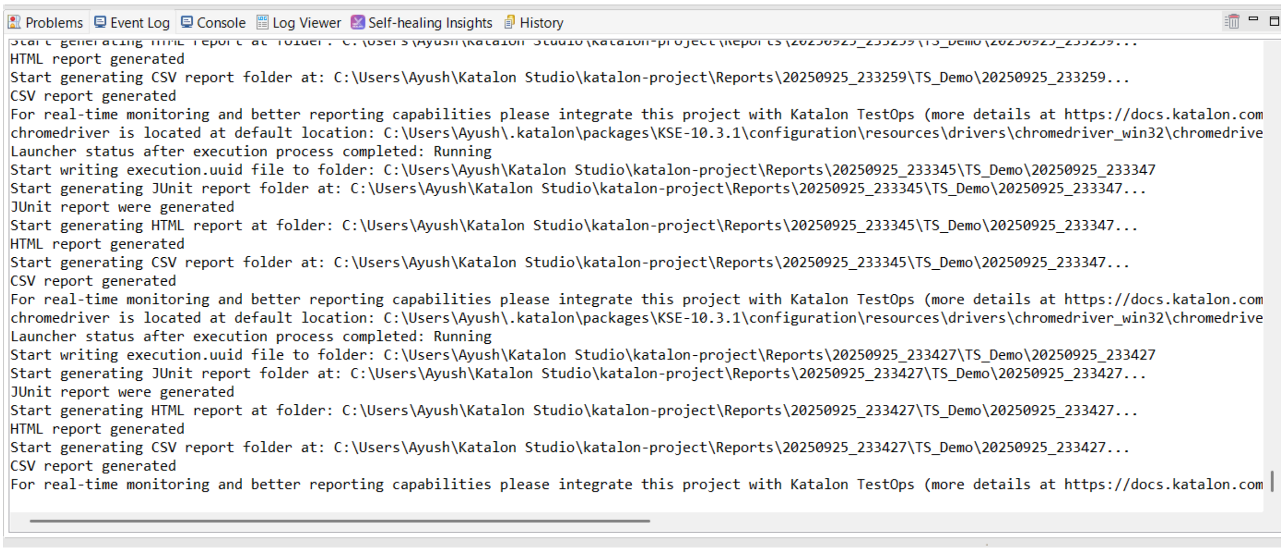
Katalon Studio Enterprise - 10.3.1-a92a7227eb - katalon-project - [Location: C:\Users\Ayush\Katalon Studio\katalon-project]

Test Cases Table

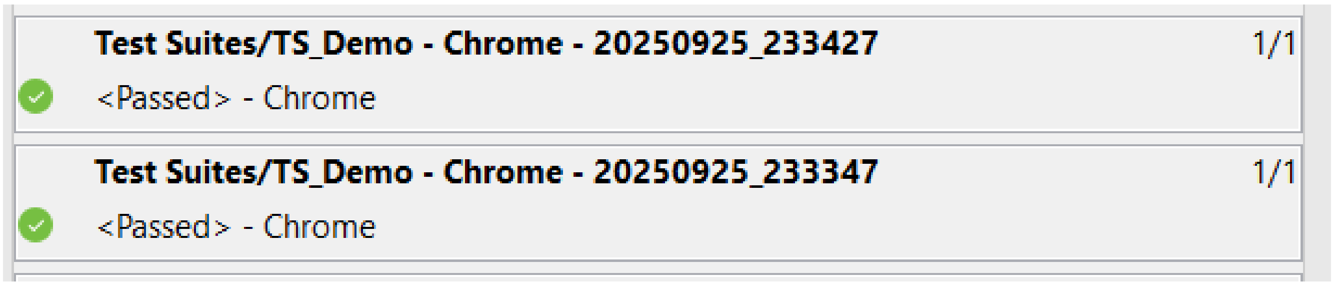
No.	Name	Resolution	Video
1	TC_OpenExample (9.449s)		

Start	25-09-2025 23:34:29	**End**	25-09-2025 23:34:40
Elapsed	10.058s		
Total TC	1		
Passed	1	**Failed**	0

```
Launcher status after execution process completed: Running
Start writing execution.uuid file to folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427
Start generating JUnit report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427...
JUnit report were generated
Start generating HTML report at folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427...
HTML report generated
Start generating CSV report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427...
CSV report generated
For real-time monitoring and better reporting capabilities please integrate this project with Katalon TestOps (more details at https://docs.katalon.com
```

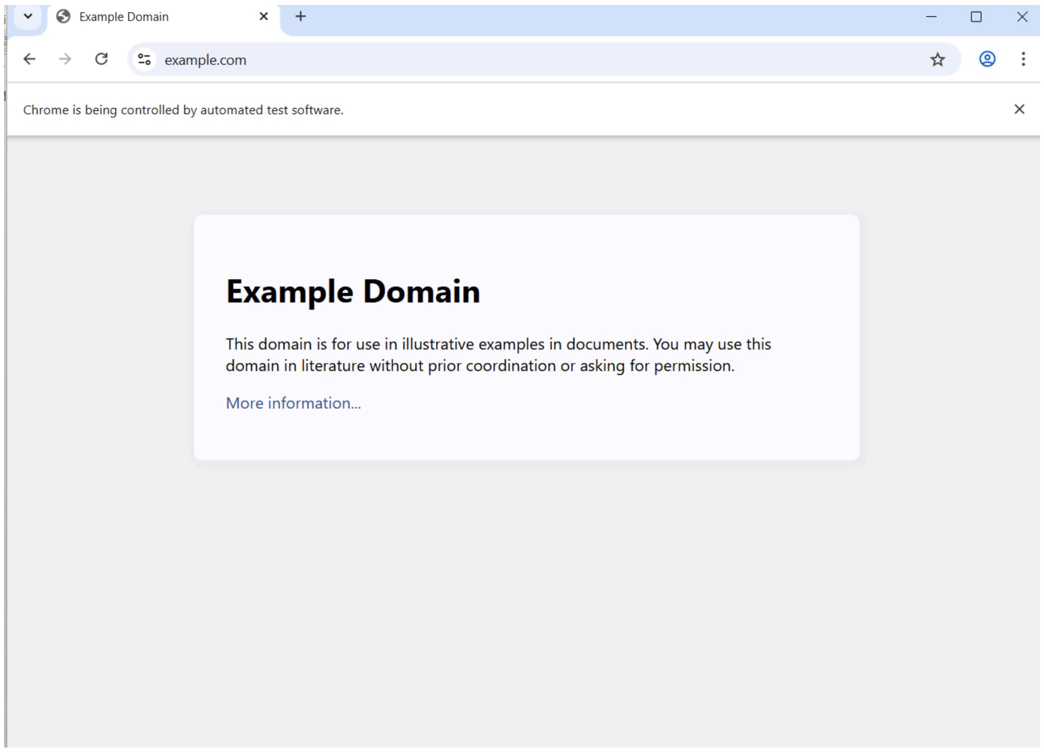


```
Start generating HTML report at folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233259\TS_Demo\20250925_233259...
HTML report generated
Start generating CSV report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233259\TS_Demo\20250925_233259...
CSV report generated
For real-time monitoring and better reporting capabilities please integrate this project with Katalon TestOps (more details at https://docs.katalon.com
chromedriver is located at default location: C:\Users\Ayush\.katalon\packages\KSE-10.3.1\configuration\resources\drivers\chromedriver_win32\chromedrive
Launcher status after execution process completed: Running
Start writing execution.uuid file to folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233345\TS_Demo\20250925_233347
Start generating JUnit report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233345\TS_Demo\20250925_233347...
JUnit report were generated
Start generating HTML report at folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233345\TS_Demo\20250925_233347...
HTML report generated
Start generating CSV report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233345\TS_Demo\20250925_233347...
CSV report generated
For real-time monitoring and better reporting capabilities please integrate this project with Katalon TestOps (more details at https://docs.katalon.com
chromedriver is located at default location: C:\Users\Ayush\.katalon\packages\KSE-10.3.1\configuration\resources\drivers\chromedriver_win32\chromedrive
Launcher status after execution process completed: Running
Start writing execution.uuid file to folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427
Start generating JUnit report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427...
JUnit report were generated
Start generating HTML report at folder: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427...
HTML report generated
Start generating CSV report folder at: C:\Users\Ayush\Katalon Studio\katalon-project\Reports\20250925_233427\TS_Demo\20250925_233427...
CSV report generated
For real-time monitoring and better reporting capabilities please integrate this project with Katalon TestOps (more details at https://docs.katalon.com
```



Test Suites/TS_Demo - Chrome - 20250925_233427 1/1
<Passed> - Chrome

Test Suites/TS_Demo - Chrome - 20250925_233347 1/1
<Passed> - Chrome

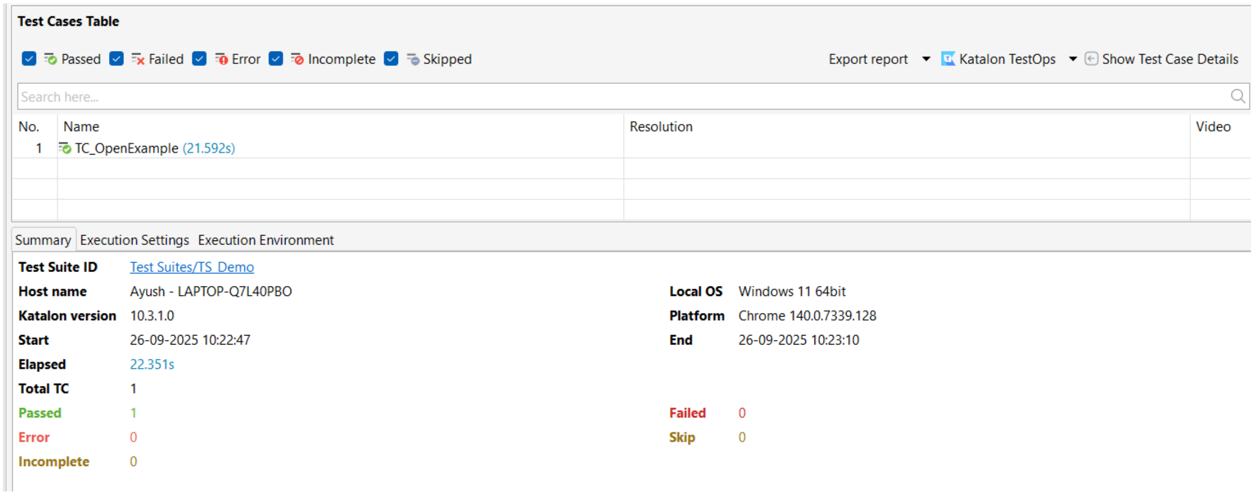


Chrome is being controlled by automated test software.

Example Domain

This domain is for use in illustrative examples in documents. You may use this domain in literature without prior coordination or asking for permission.

More information...



Test Cases Table

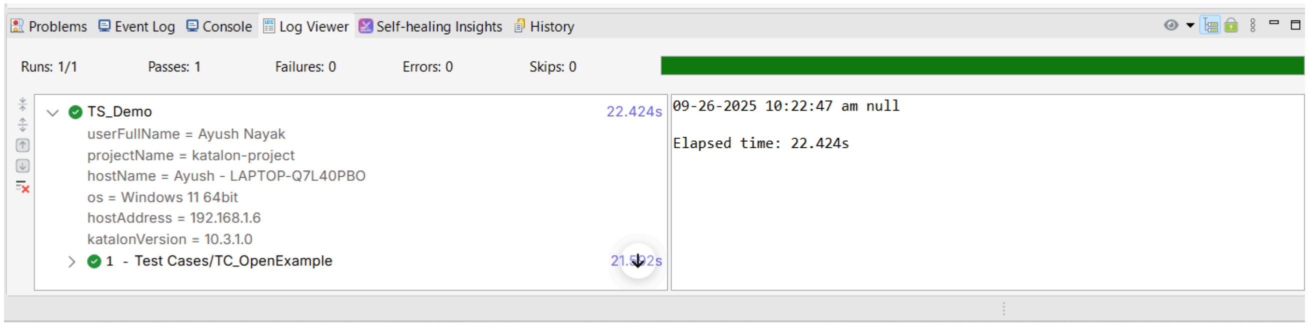
☑ Passed ☑ Failed ☑ Error ☑ Incomplete ☑ Skipped   Export report ▾ Katalon TestOps ▾ Show Test Case Details

Search here...

No.	Name	Resolution	Video
1	TC_OpenExample (21.592s)		

Summary Execution Settings Execution Environment

Test Suite ID	Test Suites/TS_Demo		
Host name	Ayush - LAPTOP-Q7L40PBO	**Local OS**	Windows 11 64bit
Katalon version	10.3.1.0	**Platform**	Chrome 140.0.7339.128
Start	26-09-2025 10:22:47	**End**	26-09-2025 10:23:10
Elapsed	22.351s		
Total TC	1		
Passed	1	**Failed**	0
Error	0	**Skip**	0
Incomplete	0		



Problems Event Log Console Log Viewer Self-healing Insights History

Runs: 1/1   Passes: 1   Failures: 0   Errors: 0   Skips: 0

```
TS_Demo                                          22.424s
    userFullName = Ayush Nayak
    projectName = katalon-project
    hostName = Ayush - LAPTOP-Q7L40PBO
    os = Windows 11 64bit
    hostAddress = 192.168.1.6
    katalonVersion = 10.3.1.0
  > 1 - Test Cases/TC_OpenExample                21.592s
```

```
09-26-2025 10:22:47 am null

Elapsed time: 22.424s
```

Conclusion:

The Katalon Studio experiment successfully demonstrated how automation can simplify and enhance the software testing process. By implementing a basic test case on `example.com`, the project showcased core capabilities such as verifying text, extracting values, capturing screenshots, and generating reports. These foundational actions highlight how even simple automation can reduce manual effort while ensuring consistency in test execution.

The integration of GitHub Actions further strengthened the experiment by enabling continuous testing in a controlled environment. Every code change pushed to the repository now triggers automated test runs, with results and artifacts made accessible to the team. This continuous integration setup not only ensures early detection of issues but also promotes transparency and collaboration among developers and testers.

Overall, the demonstration proved that Katalon Studio, combined with CI/CD, can provide an efficient, scalable, and reliable framework for test automation. It lays the groundwork for expanding test coverage to more complex scenarios, ultimately improving software quality and reducing risks. The outcome demonstrates the value of automation as a key component in modern development workflows, bridging the gap between speed and accuracy in quality assurance.