

SUMMER INTERNSHIP
On
AUTOMATED DASHBOARD PIPELINE FOR
SAOS REGRESSION INFRASTRUCTURE



By:
Name: Pragma Tripathi
Enrolment No: 00596303122
Department: Information Technology (IT-E)

AGENDA

1. ABOUT CIENA
2. INTRODUCTION
3. TECHNOLOGY USED
4. PROJECT APPROACH AND IMPLEMENTATION
5. RESULT
6. CONCLUSION

ABOUT CIENA



Fig.1. Ciena Logo

Organization Name: Ciena India Private Limited

Industry: Telecommunications

Headquarters: 7035 Ridge Road, Hanover, Maryland 21076, United States

Ciena Corporation, an American networking systems and software company, established in 1992, is a global leader in networking systems and software, **specializing in optical and routing solutions** for telecommunications and data networks. **Serving over 85% of the world's largest service providers**, Ciena's clients include AT&T, Meta, and Verizon. Ciena India, its largest R&D facility outside North America, has been recognized as **India's top optical networking company**. As of October 2022, Ciena reported revenues of \$3.63 billion and employs over 8,000 people, led by CEO Gary Smith.

INTRODUCTION

- In the fast-paced telecom industry, ensuring the reliability and scalability of network software is critical. **Ciena's Service-Aware Operating System (SAOS)** powers their global routing and switching devices, requiring continuous regression testing with every new build release.
- The Special Design & Regression Team **faced challenges as test results were scattered across multiple logs**, causing delays and limited visibility.
- To address this, my internship focused on developing an **automated live regression dashboard** that parses logs, consolidates results, and visualizes real-time test statuses. Integrated with **Jenkins CI/CD**, this system streamlined monitoring, improved transparency, and enhanced operational efficiency across Ciena's SAOS regression framework.

TECHNOLOGY USED

Linux Virtual Machine

- Linux is an open-source operating system, meaning it's free to use, modify, and distribute.
- A Linux VM was used as the primary development and testing environment for this project. It provided a stable, secure, and isolated setup to run scripts, deploy services, and manage automated pipelines efficiently within Ciena's regression infrastructure.

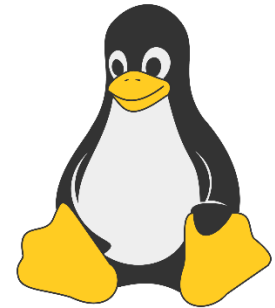


Fig 2. Linux Logo

TECHNOLOGY USED

Python Programming language

- Python is high level, easy to read, has extensive collection of libraries, cross-platform compatibility & easily integrated with CI/CD tools like Jenkins making it a perfect choice for automation pipelines.
- Python's **text-processing efficiency and file-handling capabilities** were crucial for parsing large regression log files and extracting test results.
- Python **served as the backbone** for the Flask-based web application that powered the live dashboard.
- Its **simplicity allowed for fast prototyping and iterative refinement with over 15 script versions.**



Fig 3. Python Logo

TECHNOLOGY USED

Flask

- Flask, a lightweight Python web framework, was used **to build the backend of the automated dashboard.**
- It enabled **seamless integration between the parsed regression data and the frontend interface**, allowing dynamic visualization of real-time test results.
- **Its flexibility and simplicity made it ideal for rapid development**, API creation, and efficient data handling within the SAOS regression infrastructure..



Fig 4. Flask Logo

TECHNOLOGY USED

Jenkins

- Jenkins, an open-source automation tool, was **used to trigger automated test runs** and **provide input files for the dashboard script**.
- It streamlined the testing workflow by **scheduling and executing regression tests** without manual intervention.
- By integrating Jenkins with the dashboard pipeline, the process of feeding live test data became efficient and consistent, ensuring timely and accurate updates of test results.



Fig 5. Jenkins Logo

OTHER TECHNOLOGY/ TOOLS USED




PROJECT OVERVIEW AND IMPLEMENTATION

1. Overview of SAOS Regression Infrastructure

- The SAOS Regression Infrastructure ensures the stability and reliability of Ciena's Service-Aware Operating System through continuous automated testing on Linux-based VMs, orchestrated via Jenkins.
- **Previously manual and time-consuming**, the process was transformed by the Automated Dashboard Pipeline, which now **parses, aggregates, and visualizes test results in real time**.
- This automation enhances test coverage, **accelerates feedback, reduces manual effort**, and provides transparent visibility into system performance across builds and platforms.

2. Dashboard Architecture and Design



Results for Build 01-11-01-0215 - Overall Status: **FAIL**

#	Test Suite	3948	6110	3920	5171	5164	5130
1	Onie Upgrade	PASS	PASS	PASS	PASS	PASS	PASS
2	touchtest_trex_llapi_vpws					FAIL	
3	touchtest_trex_llapi_vpls					PASS	
4	i3ypn_4nodes_test	PASS	PASS	PASS	PASS		
5	sanity_statedump						PASS

The test suite ran for 01:45:33 (HH:MM:SS) or 6,333 seconds

Fig. 6 Sample SD&R Team Dashboard
Source: Intellectual Property of Ciena India Pvt Ltd.

saos-01-11-02-0121 - Results

Results of saos-01-11-02-0121

#	Test Suite Results for evnt-saos-01-11-02-0121	3922 (10.121.182.205)	5130 (10.121.182.245)
1	sanity_generic_val_cli_test_soft_install	PASS	
2	sanity_container_sd_test_soft_install	PASS	
3	sanity_generic_val_cli_test		PASS
4	sanity_container_sd_test		PASS
	Rolled-Up Config Result	PASS	PASS
	Rolled-Up Suite Result	PASS	

Elapsed Time: 0:51:10 (HH:MM:SS)

Fig. 7 Base Dashboard with Dummy Values

3. Core Logic Development

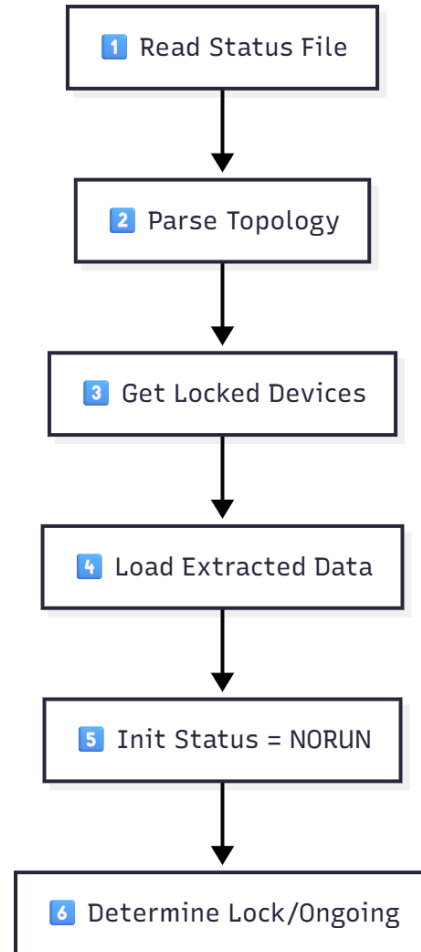


Fig 8. Data & Setup Flow

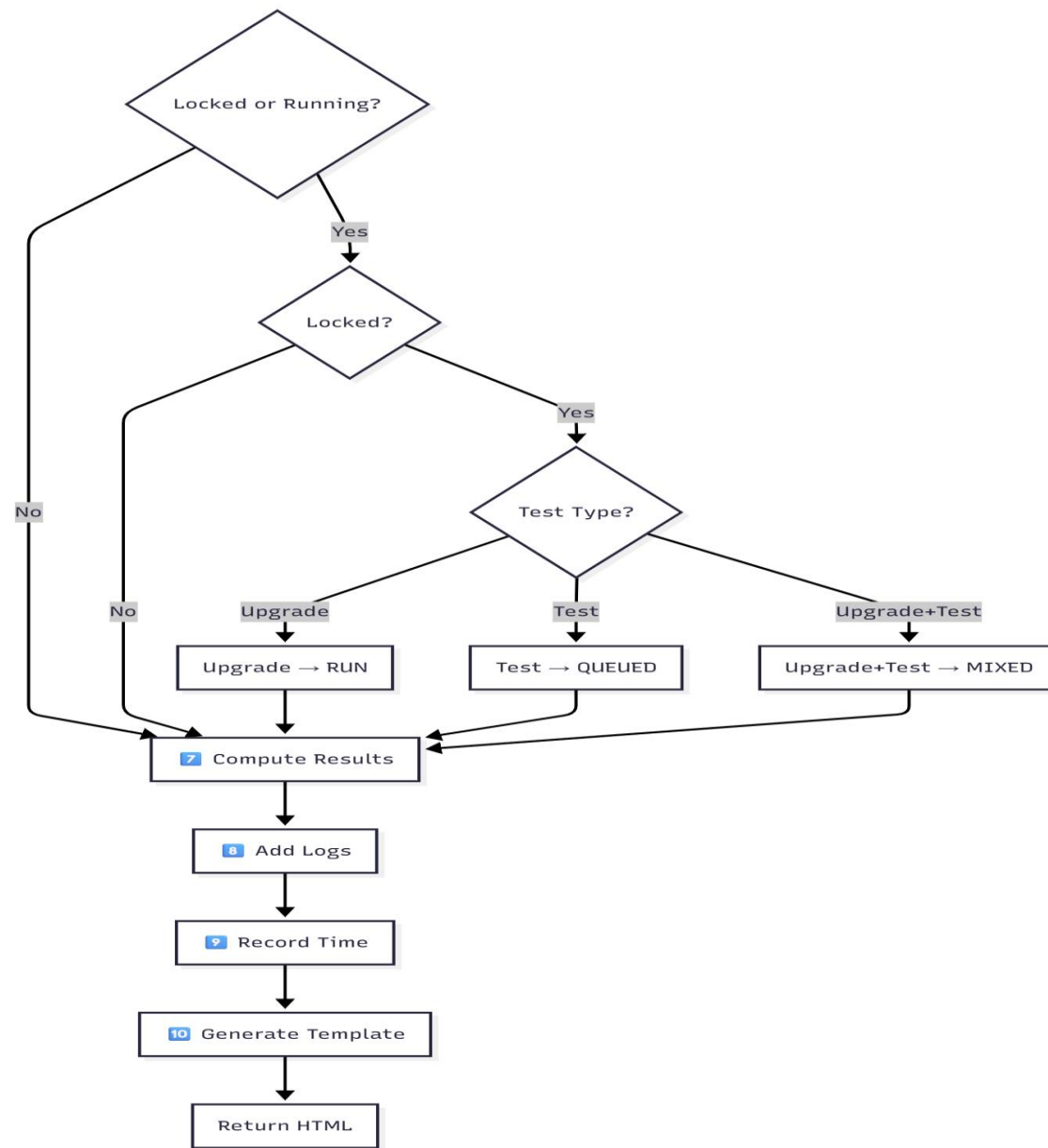


Fig 9. Status Logic & Result Flow

State	Meaning	Color
CRASH	System crash during test (Only for Onie Upgrade)	🔴 Red
FAIL	Functional failure that needs investigation	🔴 Red
PASS	Test ran and passed all checks	🟢 Light Green
WARN	Passed with warnings	🟡 Yellow
NORUN	Test not executed	🟡 Blue
QUEUED	Waiting to run	🟡 Yellow
RUNNING	Currently executing	💜 Light Cream

Fig. 10. Test Status Logic Table

Raw statuses	Result	Why
["NORUN", "NORUN"]	NORUN	All NORUN
["PASS", "NORUN"]	PASS	Only PASS and NORUN → PASS
["PASS", "PASS"]	PASS	All PASS
["FAIL", "NORUN", "PASS"]	FAIL	Any FAIL or CRASH trumps
["CRASH", "NORUN", "PASS"]	CRASH	Any CRASH trumps
["QUEUE", "NORUN"]	QUEUE	QUEUE present
["RUNNING", "PASS", "NORUN"]	RUNNING	RUNNING present
["WARN", "NORUN", "PASS"]	WARN	WARN present

Fig 11. Logic Designing for Rolled-up Config/Suite Result

Key Features Implemented

- Enabled **support for multiple projects** simultaneously through **dynamic routing**, improving scalability and flexibility.
- Implemented **live result visualization**, allowing real-time updates of test outcomes as runs progress.
- Designed and integrated **custom status logic** converting raw numeric codes into meaningful labels like *PASS*, *FAIL*, *CRASH*, *WARN*, etc.
- Applied **color-coded status indicators** for intuitive monitoring and quick status recognition.
- Added **quick access to per-status and per-platform logs**, for both **running** and **completed** test cases.
- Integrated **hyperlinked build numbers** for direct navigation to the complete test suite directory.
- Displayed **per-platform and per-testcase execution time** along with **overall execution time** in HH:MM:SS format.
- Introduced **tooltips and hover details** for enhanced clarity and improved user experience.
- Optimized **log parsing performance** for handling large-scale test runs efficiently.

RESULT

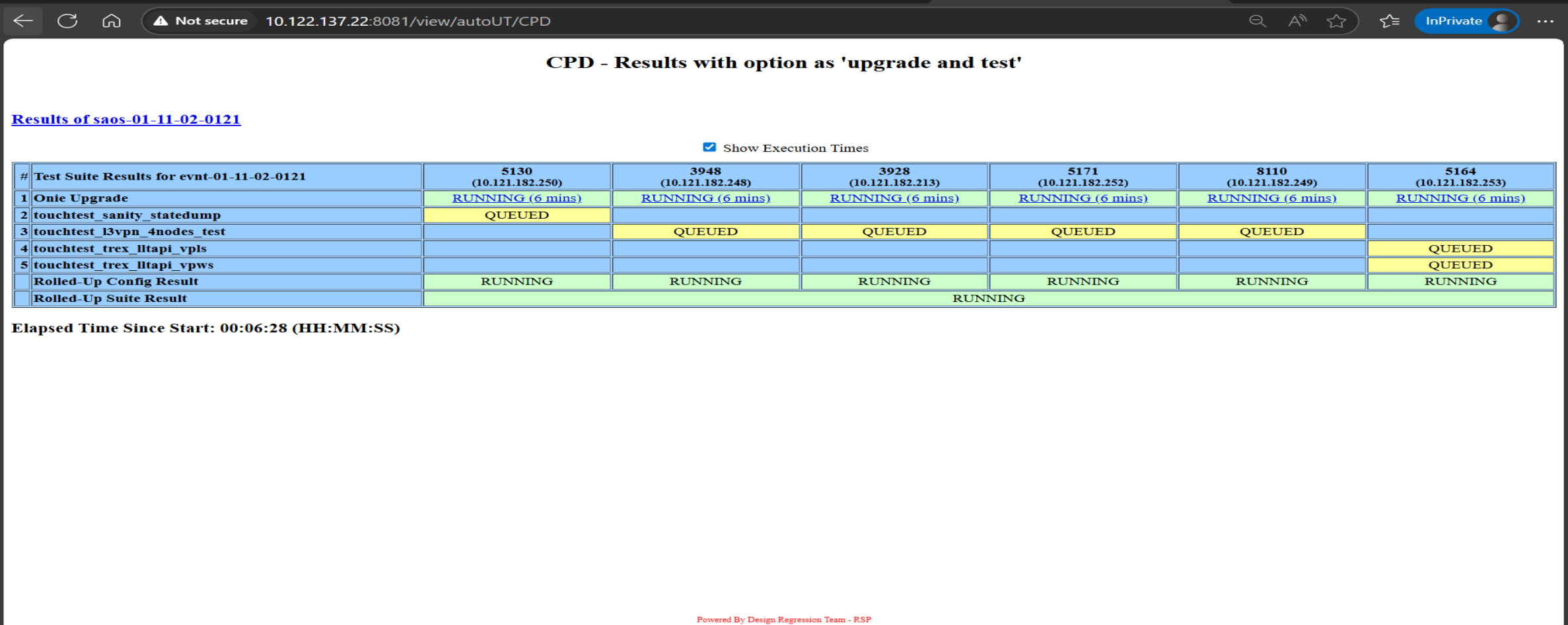


Fig 12. CPD: “Upgrade and Test” suite newly triggered

saos-01-11-02-0120 - Results												
Results of saos-01-11-02-0120												
#	Test Suite Results for evnt-saos-01-11-02-0120	3922 (10.121.182.205)	5130 (10.121.182.245)	5130 (10.121.182.138)	5164 (10.121.182.137)	5131 (10.121.182.133)	3926 (10.121.182.244)	5170 (10.121.182.82)	5170 (10.121.182.81)	5171 (10.121.182.198)	5164 (10.121.182.200)	5171 (10.121.182.197)
1	sanity_generic_val_cli_test_soft_install	FAIL										
2	sanity_container_sd_test_soft_install	PASS										
3	sanity_generic_val_cli_test		PASS									
4	sanity_container_sd_test		PASS									
5	ettp_loopback_dnx2			PASS	PASS							
6	lag_functional_stats			PASS		PASS						
7	lacp_agg_basic_tests			PASS		PASS						
8	cfm_lbm_singleTag			PASS	PASS							
9	cfm_untag_up_mep			PASS	PASS							
10	cfm_static_rmep_sanity			PASS	PASS							
11	cfm_no_modify			PASS	PASS							
12	cfm_Cos_Marking_func			PASS	PASS	PASS						
13	cfm_fp_dis_ena_func			PASS	PASS	PASS						
14	hwcfm_sanity			PASS	PASS							
15	eoam_remoteLoopback			PASS	PASS							
16	eoam_RemoteLoopbackReplace			PASS		PASS						
17	cvid_ranged_test						FAIL	FAIL				
18	system_Management								PASS			
19	https_Support								FAIL			
20	Session_local_remoteUserCoexistenceAuthSanity								PASS			
21	dot1X_Sanity								PASS	PASS		
22	gnmi_InfraTests								FAIL			
23	sanity_dhcp_lease_test										PASS	PASS
24	dhcp_ClientOptions								PASS			
25	dhcp_v6ClientOptions								PASS			
26	dhcpx4Relay_OverL3vpnRobustness								PASS	PASS	PASS	PASS
27	dhcpx6Relay_OverL3vpn6vpeRobustness								PASS	PASS	PASS	PASS
28	dhcpx4Relay_OverL3PureIpStats								FAIL			

Fig 14. Completed Test Suite Dashboard for
SAOS Build-Run

Hi All,

Please find below the URL structures for accessing results of our two scripts.

CPD (Per-Project Live Dashboard)

Kindly enter: autoUT/{PROJECT_TYPE}

URL Format: <http://10.122.137.22:8081/view/> { autoUT/{PROJECT_TYPE} }

All are running on fixed port 8081.

Examples:

- http://10.122.137.22:8081/view/autoUT/ggn_10X_test_A
- http://10.122.137.22:8081/view/autoUT/ggn_10X_test_B
- http://10.122.137.22:8081/view/autoUT/ggn_10X_test_C
- http://10.122.137.22:8081/view/autoUT/ggn_10X_test_F
- http://10.122.137.22:8081/view/autoUT/ggn_10X_test_G
- http://10.122.137.22:8081/view/autoUT/ggn_10X_quick_sanity
- http://10.122.137.22:8081/view/autoUT/ggn_10X_quick_sanity_softinstall
- http://10.122.137.22:8081/view/autoUT/ggn_10X_on_demand_test
- http://10.122.137.22:8081/view/autoUT/ggn_10X_test_VDR
- <http://10.122.137.22:8081/view/autoUT/CPD>
- <http://10.122.137.22:8081/view/autoUT/DRSanity>
- <http://10.122.137.22:8081/view/autoUT/DeviceTest>

CPD_COMMON (Per-Build Live Dashboard)

Kindly enter: {Complete Build No }

URL Format: <http://10.122.137.22:8082/view/>{Complete Build No}

Expected Build Patterns (as of now, extendable if required):

1. saos-XX-XX-XX-XXXX
2. <prefix>_saos-XX-XX-XX-XXXX
3. saos-XX-XX-XX-XXXX-<suffix>
4. CPD/saos-XX-XX-XX-XXXX/<digits>/

All are running on fixed port 8082.

Examples of actual builds:

- http://10.122.137.22:8082/view/DeviceTest_saos-01-11-02-0121 (CASE 2)
- <http://10.122.137.22:8082/view/saos-01-11-02-0117> (CASE 1)
- http://10.122.137.22:8082/view/SimDR_saos-01-11-02-0111 (CASE 2)
- <http://10.122.137.22:8082/view/saos-01-11-02-0115-nichaudh-lacp> (CASE 3)
- <http://10.122.137.22:8082/view/CPD/saos-01-11-02-0112/140820251755185758/> (CASE 4)

Please use these URLs for accessing the live dashboards.

Thanks & Regards,
Pragya Tripathi

Fig 15. Standardized URLs through dynamic routing

CONCLUSION

- The Automated Dashboard Pipeline for SAOS Regression Infrastructure has significantly improved testing efficiency by converting a manual process into a fully automated, real-time monitoring system.
- Its integration of lightweight technologies like Flask and Python ensures scalable performance across multiple projects, enhancing transparency, speed, and reliability.
- This project serves as a foundation for future advancements in intelligent test analytics and automation.

THANK YOU!