Horizon Overview

Introduction to Litmus Test

Definition of Litmus Test

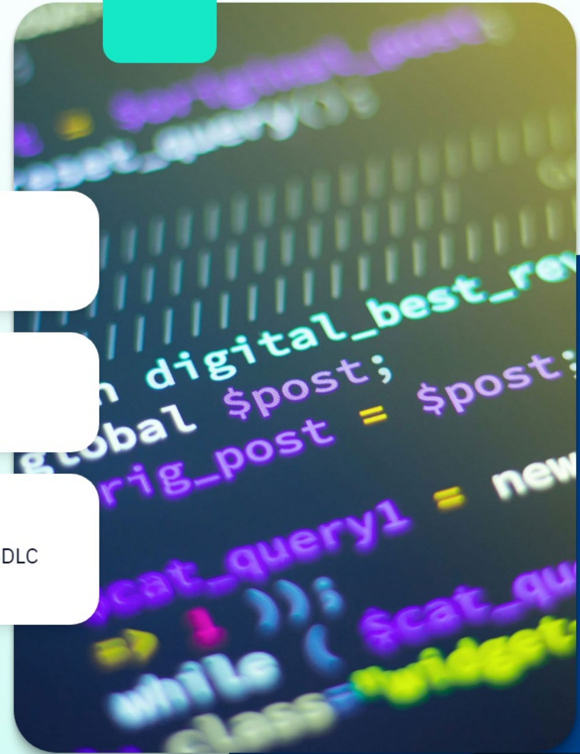
Litmus Test is a test harness tool developed as an in-house solution.

Purpose of Litmus Test

It is used for rapid testing and efficiency in the testing process.

Integration with SDLC

The tool integrates the testing process with the build and deployment process on the SDLC pipeline.



Litmus Test Functionality

01	**Purpose of Litmus Test**	Litmus Test is used to validate build and deployment events.
02	**Auto-Promotion of Code**	Helps in auto-promoting developer code into the next environments.
03	**User Interface Environment**	Provides a UI environment for users to interact with the testing process.
04	**Execution Machines Management**	Allows users to define, classify, and pool execution machines or agents for testing.

Defining Test Sets and Execution Monitoring

Definition of Test Sets

Users have the capability to define test sets that can be executed based on a schedule.

Event-Triggered Execution

Test sets can also be executed in response to build or deployment events through delivery pipelines.

User Interface Interaction

The UI interface allows users to view test summary reports, enhancing the monitoring of test execution.

Success Criteria Definition

Users can define success criteria for test execution when triggered from a build or event.

Execution Monitoring and Notifications

Litmus Test UI Overview Litmus Test UI is useful for monitoring the status of execution and execution agents. It also allows setting up notification alerts from the interface.

Components of Litmus Test

Current vs Target State of Build and Deployment

Execution Machine Management

Current State

Execution machines and agents are manually maintained.
Detection of build and deployment errors occurs post-event.

Vs

Target State with Litmus Test

Inventory and execution status are maintained automatically.
Error detection is immediate.
Errors are published to stakeholders post-execution.

Integration of Litmus with Horizon

01

Continuous Delivery Pipeline

Litmus can be integrated into the continuous delivery pipeline to enhance automation.

02

Configuration with Orchestration Tools

Litmus can be configured with build and release orchestration tools like XLR.

Litmus Execution with Deployment Events

Promotion of Development Code

01

Successful Execution

If the execution is successful, the development code is promoted to higher environments like QA, Pre-Prod environments, etc.

02

Deployment Process

This continues with deployment, and Litmus Test runs script executions, leading to code deployment from QA to UAT environments.

XLR Listening to Code Commits

XLR is integrated with the developer code repository, configured to listen for new code commits (e.g., bug fixes) through webhooks or polling.

Triggering Test Automation Pipeline

XLR Workflow in Automation Setup

Key Integration Points of XLR

01 Integration with Repositories

XLR can be integrated with both repositories based on SPK.

02 Listening for Code Commits

XLR listens to the developer code repository for new code commits.

03 Triggering Test Automation

It triggers test automation from the test automation repository.

04 Automatic Testing on Changes

Testing is automatically triggered whenever new changes are made.

Configuration Details for Integration

Integration Requirement

The test automation repository must be integrated with XLR for seamless operation.

Team Collaboration

The test automation team must provide necessary configuration details to the DevOps team.

Automation Trigger

XLR will be able to trigger and execute test automation scripts following new code commits in the developer repository.

Challenges with Deposits Automation

Single User Login Limitations

Due to single user login into the mainframe application, parallel execution is not possible from available machines or machine groups.

Advantages of CI with Litmus XLR CI/CD Pipeline

01

What is Continuous Integration?

Continuous Integration (CI) is a development practice where code changes are automatically tested.

02

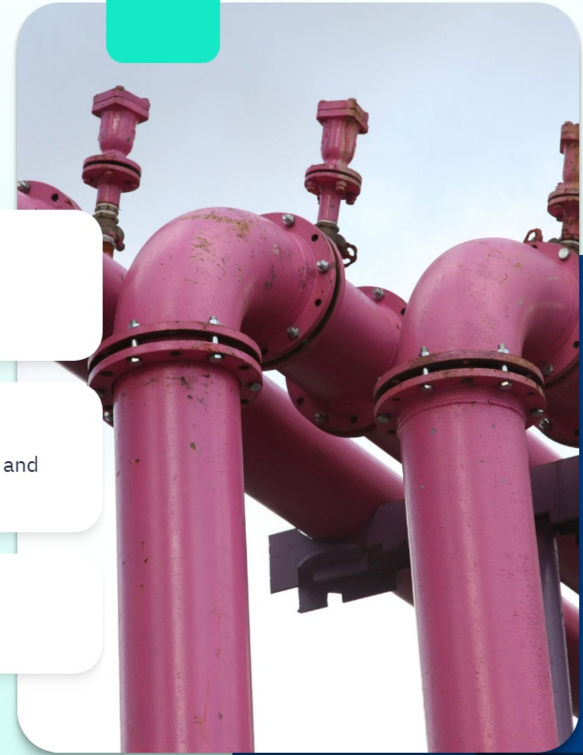
Role of XLR in CI

XLR manages changes and ensures updates in code are continuously integrated and tested early.

03

Preventing Issues

By integrating and testing early, CI prevents issues from escalating.



Reducing Manual Work with Automation Tools