Testing code with the new Gen3 testing framework

Gen3 Community Forum
6 November 2024

Agenda

- Introduction
- Announcements
- Contributor Guidelines review
- Integration testing background
- How to use the new testing framework
- Demo
- How to write your own tests
- Open Discussion

Announcements

- Next Community Forum - Gen3 Product Roadmap (January 22/23, 2025)
- Commons Security Operations Center (CSOC) Working Group
    - The Gen3 CSOC Working group will focus on topics and Gen3 improvements of interest to those organizations managing multiple Gen3 systems. A commons services operations center (CSOC) is used by organizations that run more than one Gen3 system.
    - For more information: https://gen3.org/working-groups/
    - Kick off on December 9, 2-3pm CT (December 10, 7-8am AEDT) - zoom link
    - Monthly meetings thereafter with dedicated slack channel

Gen3 Contributor Guidelines

- If you want to contribute back to Gen3 please start by reviewing our contributor guidelines: https://docs.gen3.org/docs/Contributor%20Guidelines
- Quick overview:
  - Feel free to ask questions first on slack. If it is for a complex request it is best to ask questions first.
  - If feature request or issue - post to GitHub
  - Once a PR is posted we will review and may ask questions or request edits.
  - We will try to merge quickly to source code although we may request minor or significant changes. There maybe situations where we cannot accept certain features due to security, maintenance costs, or lack of alignment with Gen3 roadmap or vision.
  - **Run existing Gen3 integration tests and provide any new tests needed for your code change**

Gen3 Integration Test Framework - Background

- General
  - Importance of integration tests for Gen3
  - Microservice architecture
  - API-first evolution

- Integration testing in Gen3
  - Original model
    - Roadmap retrospective
    - Challenges
  - Motivation for new model
    - Availability to community
    - Infrastructure agnosticism
    - Parallelization
    - Alignment with Gen3 Helm work

Gen3 Integration Test Framework - Overview

- Tools
  - **pytest**, core testing framework
  - **playwright**, for automating browser interactions
  - **gen3sdk-python**, for handling Gen3 tasks
  - **requests**, for automating API interactions
  - **allure**, for reviewing and analyzing test results
  - **xdist**, for parallel execution of tests
- Code organization
  - Tests are organized into test suites, which can run in parallel
  - Test code is organized into different directories for ease of maintenance
    - **tests**, **test_data**, **pages**, **services**, **utils**, **scripts** - explained here
  - A single conftest.py controls the test flow
  - The code for performing Gen3 operations and admin tasks is located in a single module

How to use the new testing framework

Gen3 Integration Test Framework - Usage

- Read this document for detailed instructions. TL;DR below.
- Start up a Gen3 instance (out of scope of this presentation)
  - Please see gen3-helm repo if you are using helm
- Set up prerequisites
  - Switch to $HOME and create directory named **.gen3**
  - Switch to gen3-code-vigil/gen3-integration-tests and create a file named **.env**
- Set up test users and user permissions
  - Test users and arborist permissions are specified in test_data/test_setup
- Set up test data as needed for your tests
- Run tests - using pytest commands in the document linked above
- Review results - using the allure report generated from the test run

GEN3 DATA COMMONS

Demo

How to write your own tests

Gen3 Integration Test Framework - Design Principles

- The test suites must be independent and idempotent.
- The tests must be able to run anywhere (locally / in CI) without changing test code.
- Test code is well documented.
- Test steps are documented as docstrings.
- There are no hard waits. Test should only wait for application state.
- Tests are tagged appropriately using markers, and the markers are added to pyproject.toml
- Privileged information is not logged by the tests.

Gen3 Integration Test Framework - Writing a test

- If there is a suite testing related functionality, consider adding a test to the same
- To add a new test suite, create a new module under directory **tests**
- The reusable automation code is present in one of two directories
  - **pages** - for code automating browser interactions for a given web page
  - **services** - for code automating API interactions for a given microservice
- The reusable helper code is present in the directory **utils**
- The test data is stored in test_data directory and its path object can be accessed in the tests using an import - from utils import TEST_DATA_PATH_OBJECT
- Let’s do a walkthrough of an existing test **here**

Gen3 Integration Test Framework - Final comments

- Benefits
  - Open source contributions can now include integration tests!
    - → (Open source contributions *should* now include integration tests)
  - Community members can participate in the improvement and evolution of Gen3 integration testing
  - Faster cycle time for reviewing and merging community contributions
  - Operators and developers can run integration tests faster and with greater consistency

- Next steps
  - Code to mutate helm environments for configuration testing
  - More robust data-specific testing
  - Continue helm refinement

References

- Contributor Guidelines - https://docs.gen3.org/docs/Contributor%20Guidelines
- Code Vigil GitHub Repo - https://github.com/uc-cdis/gen3-code-vigil/tree/master

Questions and Discussion

Acknowledgements

- Speakers
  - Peter Vassilatos - Center for Translational Data Science, University of Chicago
  - Hara Prasad Juvvala - Center for Translational Data Science, University of Chicago
  - Krishna Agarwal - Center for Translational Data Science, University of Chicago
- Forum Support
  - Sara Volk de Garcia - Center for Translational Data Science, University of Chicago
  - Fay Booker - Center for Translational Data Science, University of Chicago
- Gen3 Forum Steering Committee
  - Robert Grossman - Center for Translational Data Science, University of Chicago
  - Steven Manos - Australian BioCommons
  - Claire Rye - New Zealand eScience Infrastructure
  - Plamen Martinov - Open Commons Consortium
  - Michael Fitzsimons - Center for Translational Data Science, University of Chicago