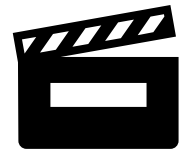


Testing



Projects in Data Sciences



Check out
my YouTube

HV Nguyen
2019



Agenda

- Application Testing Contexts: Development vs. Auditing
- Python ecosystem for Application Testing
- Unit Testing, Acceptance Testing and Integration Testing
- Demonstration

“Application” Testing: Two Contexts

Pythonistas

- *“Testing your code is very important. People are much more likely to use a project that actually works”. “A test is about the most massively useful code a hitchhiker can write.”⁽¹⁾*
- *“Obey the Testing Goat! Do Nothing Until You Have a Test”⁽²⁾*

Auditors (Regulators)

- *“Application Development Risk” associated with “Manage Change”*
- *“Control over IT process of Manage changes” ... “requirement for IT to respond to business requirements” ... “alignment with business strategy” ... “reducing solution and service delivery defects and rework”
“achieved by: (a) Define and communicate change procedures, (b) Assess, prioritize and authorize changes (c) Track status and report on changes”⁽³⁾.*



The
Testing
Goat

(1) Reitz, K. et. al. 2016 “The Hitchhiker’s Guide to Python” O’Reilly
(2) Percival, H. 2014 “Test-Driven Development with Python”, O’Reilly
(3) COBIT 5 “A16 Manage Changes, Process Description”, ISACA

Can Python “glue” these two contexts?

“Application” Testing with CI/CD in Data Sciences



CI/CD Data Sciences

- Repeatable environments
- Repeatable data
- Tested algorithms
- etc. ...

- Actionable insights
- Containerized data
- Catalogued algorithms
- etc. ...

- Productionable Insights
- Monetized services
- Insight-based decision-making
- etc. ...

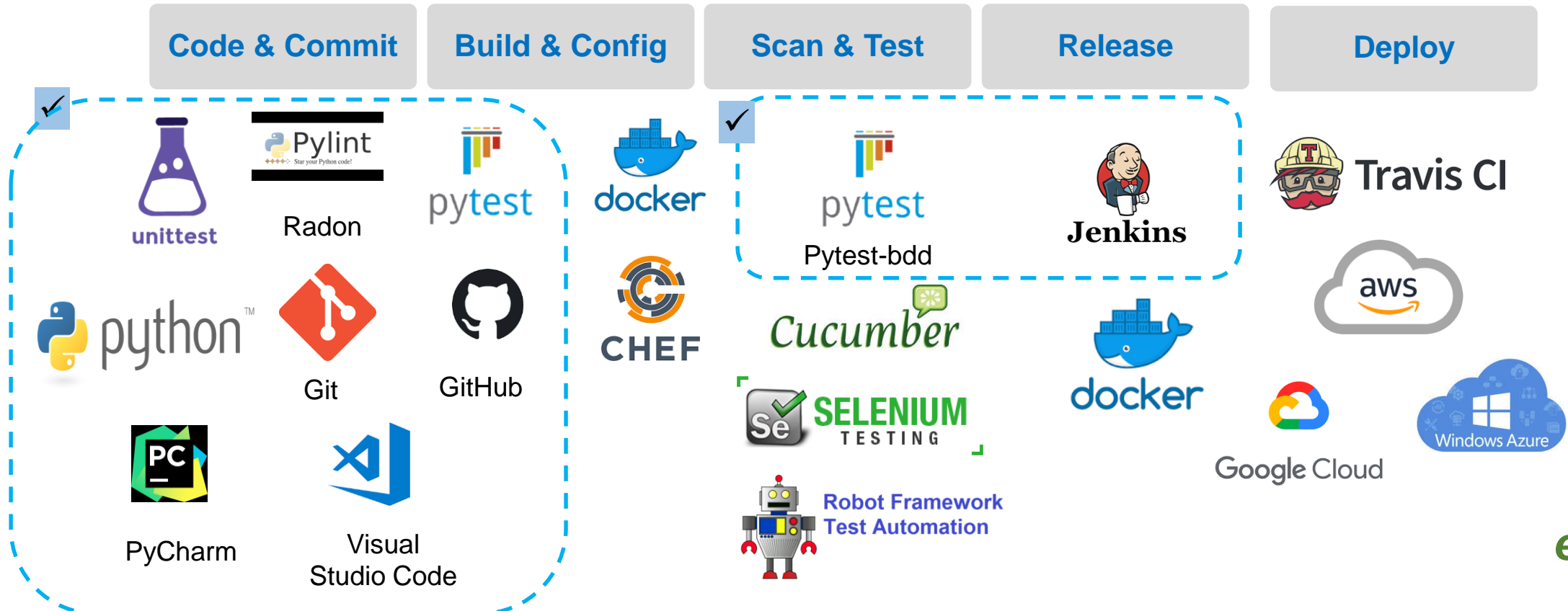
Audit/ Compliance Concerns

- Authorized Use Case
- “Vetted” source of data
- Tested algorithms
- etc. ...

- Secured repository from malicious attack
- etc. ...

- Insights traceable back to algorithms, learning data, etc.
- Engineering code not maliciously deployed
- Engineering code not deployed by unauthorized personnel
- etc. ...

“Application” Testing with Python Ecosystem



PEP-8 and Code Analysis

PEP-8	<i>Guidelines for:</i> <ul style="list-style-type: none">• <i>Code lay-out</i>• <i>Whitespace in Expressions and Statements</i>• <i>Comments</i>• <i>Version Bookkeeping</i>• <i>Naming Conventions</i>	
Developers	<ul style="list-style-type: none">• <i>Code readability and maintainability</i>• <i>Accepted Pythonistas Best Practices</i>	<ul style="list-style-type: none">• <i>Pylint</i>• <i>Radon</i>
Auditors (Regulators)	<ul style="list-style-type: none">• <i>Evidences of following Best Practices (proxy to Development Governance)</i>• <i>Bonus: development artifacts from Python development tools</i>	<i>(See Appendix A01, A02)</i>

Unit Testing

Developers	<ul style="list-style-type: none">• <i>Code coverage</i>• <i>Exceptions, errors, 'passed'</i>	<ul style="list-style-type: none">• <i>pytest</i>• <i>pytest-cov</i>
Auditors (Regulators)	<ul style="list-style-type: none">• <i>Evidences of unit testing (proxy to Development Governance)</i>• <i>Bonus:</i><ul style="list-style-type: none">• <i>Code logic + code branches</i>• <i>Test logics</i>• <i>Test results</i>• <i>Testing date</i>• <i>Automation</i>• <i>Test replication</i>	<i>(See Appendix A03, A04)</i>

Acceptance Testing

Developers	<ul style="list-style-type: none">• <i>Focus on 'behavior'</i>• <i>'Client' perspectives</i>	<ul style="list-style-type: none">• <i>pytest-bdd</i>
Auditors (Regulators)	<ul style="list-style-type: none">• <i>Evidences of User Acceptance</i>• <i>Bonus:</i><ul style="list-style-type: none">• <i>Client voice</i>• <i>Test results</i>• <i>Testing date</i>• <i>Automation</i>• <i>Test replication</i>	<i>(See Appendix A05)</i>

Integration Testing

Developers	<ul style="list-style-type: none">• <i>Repeatable integration steps</i>• <i>'Hands-off'</i>• <i>Control over deployment</i>	<ul style="list-style-type: none">• <i>jenkins</i>
Auditors (Regulators)	<ul style="list-style-type: none">• <i>Evidences of Deployment</i>• <i>Evidences of Segregation of Duties (developers not tampering with production)</i>• <i>Bonus:</i><ul style="list-style-type: none">• <i>Deployment logic</i>• <i>Deployment results</i>• <i>Deployment date</i>• <i>Automation</i>• <i>Deployment replication</i>	<i>(See Appendix A06, A07, A08)</i>

What Next?

- Considerations for Machine Learning
 - ML is (especially) hard
 - (Seemingly) random range of “almost”-correct equilibrium (end) states
 - Solutions? Write Tests
 - Subtle, Case-specific Problems? Write (more) Tests
 - Generative Learner? Generative training/test algo/data
 - Discriminative Learner? SGD (gradient step, loss min., etc.)
 - Approaches
 - Model Assertions (running and training time)
 - Runtime monitoring, Corrective action, Active Learning, Weak Supervision
 - Debugging Tasks
 - Classification (training data), Root cause (Pearl’s Probability of Sufficiency), etc.

References

1. Cadamuro G., et. al. (Preprint) “Debugging Machine Learning Models”, University of Washington
2. Chakarov A. et.al. 2016 “Debugging Machine Learning Tasks”, University of Colorado
3. Cohen W. (Presentation) “Debugging Machine Learning Algorithms” CMU (<http://www.cs.cmu.edu/~wcohen/10-605/debugging-tips.pdf>)
4. Kang D. et.al. (Preprint) “Model Assertions for Debugging Machine Learning”, Stanford DAWN Project

Question & Answer



I'll try my best...

References

Documentation

- [COBIT 5 “A16 Manage Changes, Process Description”, ISACA](#)
- [Jenkins documentation](#)
- [luigi test github](#)
- [PEP-008 Style Guide for Python Code](#)
- [pylint homepage](#)
- [pytest documentation](#)
- [pytest-bdd documentation](#)
- [Radon documentation](#)
- [unittest.mock Object Library](#)

Literature

- [Okken, B. 2017 “Python Testing with pytest” The Pragmatic Programmer](#)
- [Percival, H. 2014 “Test-Driven Development with Python” O’Reilly](#)
- [Reitz, K. et. al. 2016 “The Hitchhiker’s Guide to Python” O’Reilly](#)
- [Sale, D. 2015 “Testing Python Unit Testing, TDD, BDD” Wiley](#)

Appendices

A00 PSET-4-5 GITHUB Repository

README.md

Pset 4.5

So you are happy with your midterm test and you have finished (as usual) your problem set ("pset") weeks ahead of time. You fell in love with your achievement in the pset (and the mastery of new skills: Luigi, Pytorch, rendering the picture of your pre-school, etc.). You are now wondering what to do next with your free time. You are in luck!!! This pset will give you the chance to re-live all the above happy memory, while letting you indulge in your number one past time: TESTING. This exercise will give you the opportunities to refresh your knowledge of testing by focusing on Python and its test ecosystem including pytest, mock, tox and more.

Table of Contents *generated with DocToc*

- Setup
 - Render your pset 4 repo
 - Installations
 - AWS credentials
 - Styling Code
 - Fixing their work
 - Pre-Trained Model & Input Content Image
 - Limit builds on Travis
- Problems (45 points)
 - A new atomic write (10 points)
 - External Tasks (5 points)
 - Copying S3 files locally (15 points)
 - Stylizing (15 points)
 - Option A) ExternalProgramTask
 - Option B) Direct python
 - Running it
 - Your Own Image (points in quiz)

Testing in Data Sciences

Assignment Objectives

This assignment has the following objectives:

1. Refresh for students the key concepts of testing including unit testing, integration testing and acceptance testing
2. Get the students a summary of the key concepts of testing in the specific context of Python

```
— setup.cfg
— setup.py
— src
  — neural_style
    — __init__.py
    — __main__.py
    — neural_style.py
    — transformer_net.py
    — utils.py
    — vgg.py
  — pset_4
    — cli.py
    — cli.pyc
    — __init__.py
    — __init__.pyc
    — __main__.py
    — tasks
      — data.py
      — __init__.py
      — __main__.py
      — stylize.py
  — pset_utils
    — luigi
      — __init__.py
      — target.py
    — vio
      — cli.py
      — hash_str.py
      — __init__.py
      — io.py
      — load_data.py
      — __main__.py
      — similarity.py
      — WordEmbedding.py
— tests
  — features
    — pset_4.feature
  — test_pset_4.py
  — test_steps
    — pset_4_Stylize.py
```

Local repository

<https://github.com/nhvinh118/pset-4-5>

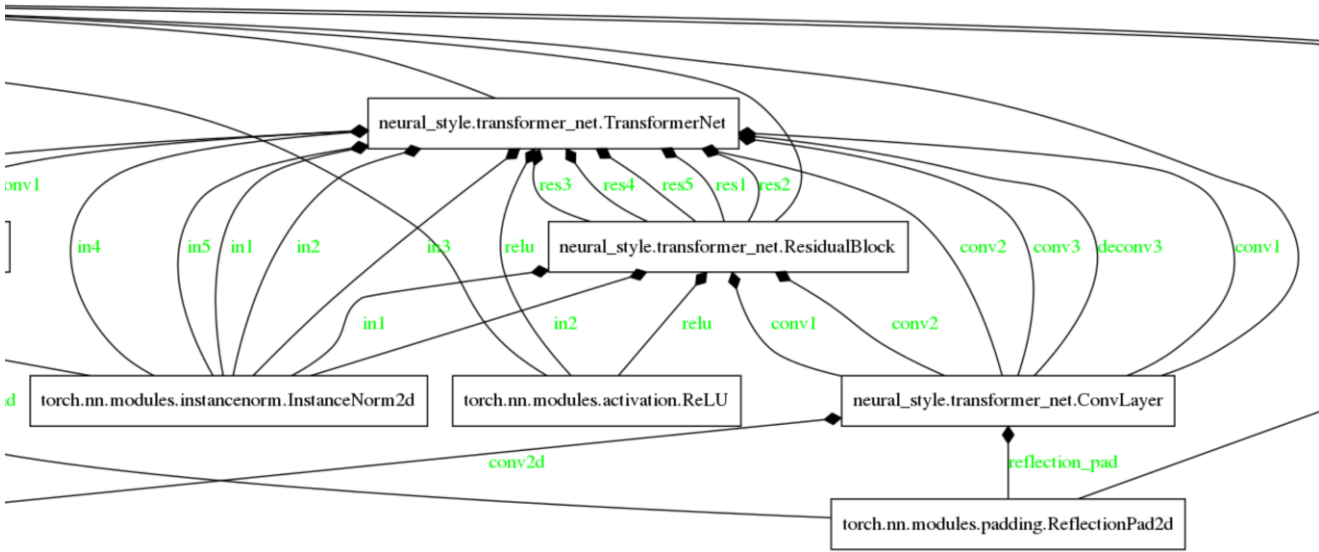
A01 Code Analysis Pylint

```
***** Module pset_4
src/pset_4/__init__.py:1:0: C0111: Missing module docstring (missing-docstring)
***** Module pset_4.__main__
src/pset_4/__main__.py:9:0: C0303: Trailing whitespace (trailing-whitespace)
***** Module pset_4.cli
src/pset_4/cli.py:13:0: C0303: Trailing whitespace (trailing-whitespace)
src/pset_4/cli.py:14:0: W0312: Found indentation with tabs instead of spaces (mixed-indentation)
src/pset_4/cli.py:15:0: W0312: Found indentation with tabs instead of spaces (mixed-indentation)
src/pset_4/cli.py:15:67: C0326: No space allowed around keyword argument assignment
      main_arg_parser.add_argument("--content-image", type=str, default = '')
                          ^ (bad-whitespace)

src/pset_4/tasks/stylize.py:21:0: C0111: Missing class docstring (missing-docstring)
src/pset_4/tasks/stylize.py:26:1: W0231: __init__ method from base class 'Task' is not called (s
src/pset_4/tasks/stylize.py:39:33: E0602: Undefined variable 'args' (undefined-variable)
src/pset_4/tasks/stylize.py:40:2: C0103: Variable name "LocalTargetSimg" doesn't conform to snake
src/pset_4/tasks/stylize.py:48:3: E0602: Undefined variable 'args' (undefined-variable)
src/pset_4/tasks/stylize.py:49:3: E0602: Undefined variable 'args' (undefined-variable)
src/pset_4/tasks/stylize.py:50:3: E0602: Undefined variable 'args' (undefined-variable)
src/pset_4/tasks/stylize.py:51:3: E0602: Undefined variable 'args' (undefined-variable)
src/pset_4/tasks/stylize.py:53:11: E0602: Undefined variable 'args' (undefined-variable)
src/pset_4/tasks/stylize.py:11:0: W0611: Unused import argparse (unused-import)
src/pset_4/tasks/stylize.py:14:0: W0611: Unused import luigi (unused-import)
***** Module pset_4.tasks.__main__
src/pset_4/tasks/__main__.py:9:0: C0303: Trailing whitespace (trailing-whitespace)
***** Module pset_4.tasks
src/pset_4/tasks/__init__.py:1:0: C0111: Missing module docstring (missing-docstring)

-----
Your code has been rated at -6.05/10 (previous run: -6.05/10, +0.00)
```

pylint report



pylint-generated
class diagram

A02 Code Analysis Radon

```
src/pset_utils/luigi/target.py
M 56:1 BaseAtomicProviderLocalTarget.open - A (4)
C 41:0 BaseAtomicProviderLocalTarget - A (3)
C 16:0 suffix_preserving_atomic_file - A (2)
M 27:1 suffix_preserving_atomic_file.generate_tmp_path - A (2)
M 50:1 BaseAtomicProviderLocalTarget.__init__ - A (2)
M 19:1 suffix_preserving_atomic_file.__init__ - A (1)
M 22:1 suffix_preserving_atomic_file.__enter__ - A (1)
M 75:1 BaseAtomicProviderLocalTarget.tmp_path - A (1)
M 82:1 BaseAtomicProviderLocalTarget.tmp_path - A (1)
C 89:0 SuffixPreservingLocalTarget - A (1)
```

Radon Cyclomatic Complexity (CC)

CC score	Rank	Risk
1 - 5	A	low - simple block
6 - 10	B	low - well structured and stable block
11 - 20	C	moderate - slightly complex block
21 - 30	D	more than moderate - more complex block
31 - 40	E	high - complex block, alarming
41+	F	very high - error-prone, unstable block

Block type	Letter
Function	F
Method	M
Class	C

```
src/neural_style/transformer_net.py
LOC: 99
LLOC: 74
SLOC: 72
Comments: 4
Single comments: 4
Multi: 9
Blank: 14
- Comment Stats
  (C % L): 4%
  (C % S): 6%
  (C + M % L): 13%
```

```
src/neural_style/transformer_net.py - A (70.85)
src/neural_style/utils.py - A (65.61)
src/neural_style/__main__.py - A (81.17)
src/neural_style/vgg.py - A (63.27)
src/neural_style/neural_style.py - A (37.59)
src/neural_style/__init__.py - A (100.00)
src/pset_4/__main__.py - A (81.17)
src/pset_4/cli.py - A (94.10)
src/pset_4/__init__.py - A (100.00)
src/pset_4/tasks/data.py - A (74.82)
src/pset_4/tasks/stylize.py - A (86.41)
src/pset_4/tasks/__main__.py - A (81.17)
src/pset_4/tasks/__init__.py - A (100.00)
src/pset_utils/luigi/target.py - A (70.42)
src/pset_utils/luigi/__init__.py - A (100.00)
src/pset_utils/vio/similarity.py - A (91.44)
src/pset_utils/vio/__main__.py - A (81.17)
src/pset_utils/vio/WordEmbedding.py - A (77.96)
src/pset_utils/vio/hash_str.py - A (92.92)
src/pset_utils/vio/cli.py - A (100.00)
src/pset_utils/vio/io.py - A (97.11)
src/pset_utils/vio/load_data.py - A (82.82)
src/pset_utils/vio/__init__.py - A (100.00)
```

Radon Raw

- **LOC**: the total number of lines of code
- **LLOC**: the number of logical lines of code
- **SLOC**: the number of source lines of code - not necessarily corresponding to the **LLOC**
- **comments**: the number of Python comment lines (i.e. only single-line comments #)
- **multi**: the number of lines representing multi-line strings
- **blank**: the number of blank lines (or whitespace-only ones)

Radon Maintainability Index (MI)

MI score	Rank	Maintainability
100 - 20	A	Very high
19 - 10	B	Medium
9 - 0	C	Extremely low

A03 Unit Test Code

```
# -*- coding: utf-8 -*-
""" Module to illustrate the unit testing of luigi tasks
Source: https://github.com/spotify/luigi/tree/master/test
"""
import os
import tempfile
import unittest
import functools
import itertools

import boto3
from boto.s3 import key

from mock import patch
from moto import mock_s3, mock_sts

from luigi.mock import MockTarget
from luigi.contrib.s3 import (DeprecatedBotoClientException, FileNotFoundException,
                             from luigi import six

from pset_4.tasks.data import ContentImage, SavedModel, CopyS3ImageLocally, CopyS3ModelLocally
from pset_4.tasks.styleize import Stylize
from pset_4.cli import main as climain
from pset_utils.luigi.target import suffix_preserving_atomic_file, \

AWS_ACCESS_KEY = "XXXXXXXXXXXXXXXXXXXX"
AWS_SECRET_KEY = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"

def create_bucket():

class with_config(object):

class ContentImageTest(unittest.TestCase, ContentImage):

class SavedModelTest(unittest.TestCase, SavedModel):

class CopyS3ImageLocallyTest(unittest.TestCase, CopyS3ImageLocally):
    # self.assertTrue(s3_client.exists('s3://psetbucket/tempfile'))
    # self.assertFalse(s3_client.exists('s3://psetbucket/temp'))

    # s3_client.put(self.tempFilePath, 's3://psetbucket/tempdir0_$folder$')
    # self.assertTrue(s3_client.exists('s3://psetbucket/tempdir0'))
    #
    # s3_client.put(self.tempFilePath, 's3://psetbucket/tempdir1/')
    # self.assertTrue(s3_client.exists('s3://psetbucket/tempdir1'))
    #
    # s3_client.put(self.tempFilePath, 's3://psetbucket/tempdir2/subdir')
    # self.assertTrue(s3_client.exists('s3://psetbucket/tempdir2'))
    # self.assertFalse(s3_client.exists('s3://psetbucket/tempdir'))

class CopyS3ModelLocallyTest(unittest.TestCase, CopyS3ModelLocally):

class StylizeTest(unittest.TestCase, Stylize):

class SuffixPreservingTest(unittest.TestCase, suffix_preserving_atomic_file):

class BaseAtomicTest(unittest.TestCase, BaseAtomicProviderLocalTarget):

class SuffixPreserveTest(unittest.TestCase, SuffixPreservingLocalTarget):

def test_climain():

if __name__ == '__main__':
```

unittest

AWS boto,
boto3

Luigi
DAG

Function
coverage

test_pset4.py

```
class ContentImageTest(unittest.TestCase, ContentImage):
    """ Test class ContentImage """

    def setUp(self):
        f = tempfile.NamedTemporaryFile(mode='wb', delete=False)
        self.tempFileContents = (
            self.tempFilePath = f.name
            f.write(self.tempFileContents)
            f.close()
            self.addCleanup(os.remove, self.tempFilePath)

        self.mock_s3 = mock_s3()
        self.mock_s3.start()
        self.addCleanup(self.mock_s3.stop)

    def tearDown(self):
        pass

    def Output(self):
        return MockTarget("output Image")

    def create_target(self, format=None, **kwargs):
        client = S3Client(AWS_ACCESS_KEY, AWS_SECRET_KEY)
        create_bucket()
        return S3Target('s3://psetbucket/luigi.jpg', client=client, format=format, **kwargs)

    def test_read(self):
        # Test read file when file exists
        client = S3Client(AWS_ACCESS_KEY, AWS_SECRET_KEY)
        create_bucket()
        client.put(self.tempFilePath, 's3://psetbucket/tempfile')
        t = S3Target('s3://psetbucket/tempfile', client=client)
        read_file = t.open()
        file_str = read_file.read()
        self.assertEqual(self.tempFileContents, file_str.encode('utf-8'))

    def test_read_no_file(self):
        # Test read file when file does not exist
        t = self.create_target()
        self.assertRaises(FileNotFoundException, t.open)

    def test_read_iterator_long(self):
        # Test iteration - write a file that is 5X the boto buffersize
        old_buffer = key.Key.BufferSize
        key.Key.BufferSize = 2
        try:
            finally:

                self.assertEqual(3, len(lines))
                self.assertEqual(firstline, lines[0])
                self.assertEqual(secondline, lines[1])
                self.assertEqual(thirdline, lines[2])

    def test_get_path(self):
        # Test get path
        t = self.create_target()
        path = t.path
        self.assertEqual('s3://psetbucket/luigi.jpg', path)
```

TmpFile
setup

Mock
S3

Mock
Target

Test
S3Client

Test
S3Target

A04 Unit Test pytest report

pytest_pset45.html

Report generated on 04-May-2019 at 13:03:47 by [pytest-html](#) v1.20.0

Environment

Packages	{'pytest': '4.4.1', 'py': '1.8.0', 'pluggy': '0.9.0'}
Platform	Linux-4.15.0-48-generic-x86_64-with-debian-buster-sid
Plugins	{'metadata': '1.8.0', 'html': '1.20.0', 'cov': '2.7.1'}
Python	3.7.2

Summary

6 tests ran in 2.29 seconds.

(Un)check the boxes to filter the results.

☒ 5 passed, ☒ 0 skipped, ☒ 1 failed, ☒ 4 errors, ☒ 0 expected failures, ☒ 0 unexpected passes

Results

[Show all details](#) / [Hide all details](#)

Result	Test
Error (show details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_exists::setup
Error (show details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_get_key::setup
Error (show details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_init_with_config::setup
Error (show details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_init_without_init_or_config::setup
Failed (hide details)	tests/test_pset_4.py::test_climain
<pre>tests/test_pset_4.py:355: in test_climain climain({}) src/pset_4/cli.py:23: in main args = main_arg_parser.parse_args() /usr/local/lib/python3.7/argparse.py:1752: in parse_args self.error(msg % ' '.join(argv)) /usr/local/lib/python3.7/argparse.py:2501: in error self.exit(2, _('%(prog)s: error: %(message)s\n') % args) /usr/local/lib/python3.7/argparse.py:2488: in exit _sys.exit(status) E SystemExit: 2 ----- Captured stderr call ----- usage: pytest [-h] [--content-image CONTENT_IMAGE] [--content-scale CONTENT_SCALE] [--output-image OUTPUT_IMAGE] [--cuda CUDA] [--export_onnx EXPORT_ONNX] [-i IMAGE] [-m MODEL] pytest: error: unrecognized arguments: --html=./reports/pytest_pset45.html</pre>	
Passed (show details)	tests/test_pset_4.py::test_pset_4.with_config
Passed (show details)	tests/test_pset_4.py::ContentImageTest::test_get_path
Passed (show details)	tests/test_pset_4.py::ContentImageTest::test_read
Passed (show details)	tests/test_pset_4.py::ContentImageTest::test_read_iterator_long
Passed (show details)	tests/test_pset_4.py::ContentImageTest::test_read_no_file

Date report generated

Environment

Summary

'Error' section

'Failed' section

'Passed' section

pytest_pset45.html

Report generated on 04-May-2019 at 13:03:47 by [pytest-html](#) v1.20.0

Environment

Packages	{'pytest': '4.4.1', 'py': '1.8.0', 'pluggy': '0.9.0'}
Platform	Linux-4.15.0-48-generic-x86_64-with-debian-buster-sid
Plugins	{'metadata': '1.8.0', 'html': '1.20.0', 'cov': '2.7.1'}
Python	3.7.2

Summary

6 tests ran in 2.29 seconds.

(Un)check the boxes to filter the results.

☒ 5 passed, ☒ 0 skipped, ☒ 1 failed, ☒ 4 errors, ☒ 0 expected failures, ☒ 0 unexpected passes

Results

[Show all details](#) / [Hide all details](#)

Result	Test
Error (hide details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_exists::setup
<pre>../../../../local/share/venv/pset-4-5-7H9seXuk/lib/python3.7/site-packages/luigi/task_register.py:88: in __call__ param_values = cls.get_param_values(params, args, kwargs) ../../../../local/share/venv/pset-4-5-7H9seXuk/lib/python3.7/site-packages/luigi/task.py:405: in get_param_values raise parameter.UnknownParameterException('%s: takes at most %d parameters (%d given)' % (exc_desc, len(positional_params), len(args)) E luigi.parameter.UnknownParameterException: CopyS3ImageLocallyTest[args=('test_exists',), kwargs={}]: takes at most 0 parameters (1 given)</pre>	
Error (hide details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_get_key::setup
<pre>../../../../local/share/venv/pset-4-5-7H9seXuk/lib/python3.7/site-packages/luigi/task_register.py:88: in __call__ param_values = cls.get_param_values(params, args, kwargs) ../../../../local/share/venv/pset-4-5-7H9seXuk/lib/python3.7/site-packages/luigi/task.py:405: in get_param_values raise parameter.UnknownParameterException('%s: takes at most %d parameters (%d given)' % (exc_desc, len(positional_params), len(args)) E luigi.parameter.UnknownParameterException: CopyS3ImageLocallyTest[args=('test_get_key',), kwargs={}]: takes at most 0 parameters (1 given)</pre>	
Error (show details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_init_with_config::setup
Error (show details)	tests/test_pset_4.py::CopyS3ImageLocallyTest::test_init_without_init_or_config::setup

'Error' details

pytest report (html)

A05 “Acceptance” Test pytest-bdd

Folders containing
pytest-bdd
codes

```
# Created by hoang at 5/4/19
Feature: Stylize
  As a graphic designer,
  I want to stylize a given picture,
  So I can embellish this picture,
  Using the graphic pattern from a model that my client has chosen.

Scenario: Picture fetching
  Given I have the picture in S3
  And I have provided AWS_ACCESS_KEY
  And I have provide AWS_SECRET_KEY
  Then I can fetch the picture from S3

Scenario: Picture saving
  Given I have fetched the picture from S3
  And I have created the local folder 'data'
  Then I can save the picture in 'data'

Scenario: Model fetching
  Given I have the model in S3
  And I have provided AWS_ACCESS_KEY
  And I have provide AWS_SECRET_KEY
  Then I can fetch the model from S3

Scenario: Model saving
  Given I have fetched the model from S3
  And I have created the local folder 'data'
  Then I can save the model in 'data'

Scenario: Picture stylizing
  Given I have saved the picture in 'data'
  And I have saved the model in 'data'
  Then I can stylize the picture using the model
```

```
from pytest_bdd import scenario, given, when, then

@scenario("stylize", "Stylize a picture")
def test_stylize_feature():
    pass

@given('I have the picture in S3')
def test_exists_picture():
    return 'picture_exists'

@given('I have fetched the picture from S3')
def test_fetched_picture():
    return 'picture_fetched'

@given('I have the model in S3')
def test_exists_model():
    return 'model_exists'

@given('I have fetched the model from S3')
def test_fetched_model():
    return 'model_fetched'

@then("I can fetch the picture from S3")
def step_impl():
    raise NotImplementedError(u'STEP: Then I can fetch the picture from S3')

@then("I can save the picture in 'data'")
def step_impl():
    raise NotImplementedError(u'STEP: Then I can save the picture in \'data\'')

@then("I can fetch the model from S3")
def step_impl():
    raise NotImplementedError(u'STEP: Then I can fetch the model from S3')

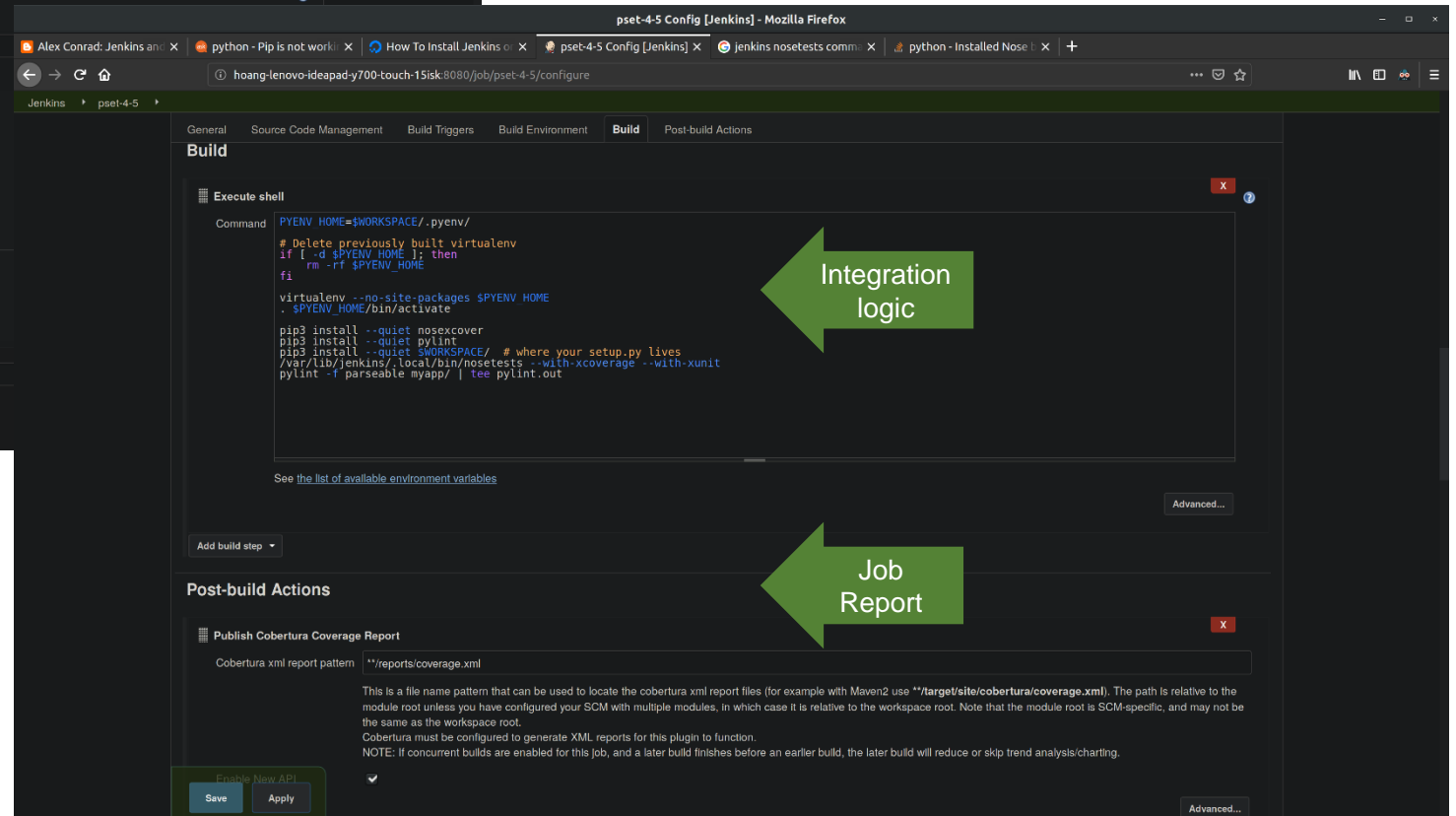
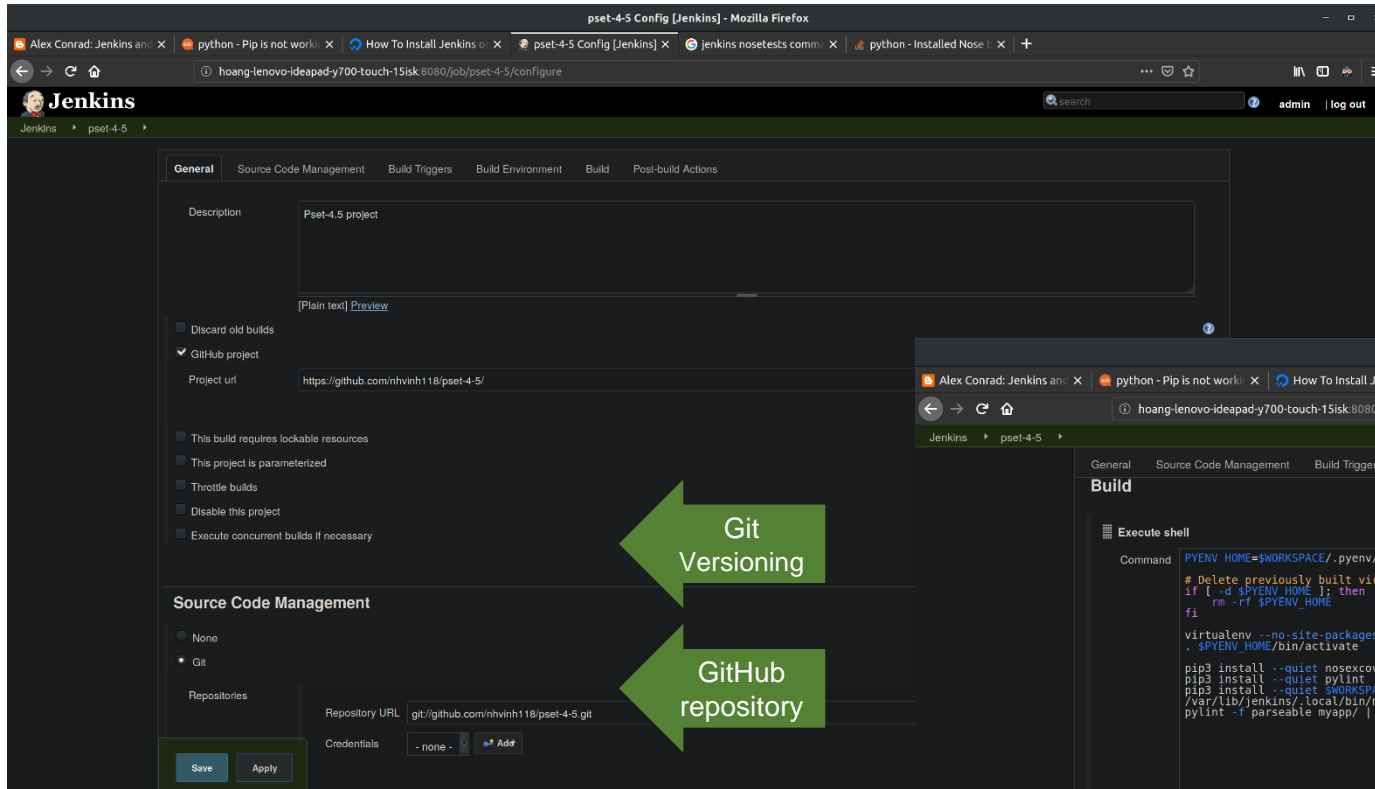
@then("I can save the model in 'data'")
def step_impl():
    raise NotImplementedError(u'STEP: Then I can save the model in \'data\'')

@then("I can stylize the picture using the model")
def step_impl():
    raise NotImplementedError(u'STEP: Then I can stylize the picture using the model')
```

“Features”: English like
description of the functionality

“Test Steps”: Translation of
features into testable code

A06 “Integration” Jenkins Configuration



A07 “Integration” Jenkins Dashboard & Workspace

The screenshot shows the Jenkins dashboard in a web browser. The top navigation bar includes links for 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Lockable Resources', 'Credentials', and 'New View'. The main content area displays a table of builds in the queue, with columns for 'S', 'W', 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. A large green arrow points to the 'pset-4-5' build. Below the table, there are sections for 'Build Queue' and 'Build Executor Status'. On the right, a sidebar shows the 'Workspace of pset-4-5' with a file tree view.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		pset-4-5	N/A	16 min - #12	8.5 sec

Icon: S M L

Legend: RSS for all RSS for failures RSS for just latest builds

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 Idle

Workspace of pset-4-5

- git
- pyenv
- vscode
- cl
- docs
- reports
- src
- tests
- bumpversion.cfg
- cookiecutter
- coveragerc
- editorconfig
- gitignore

Jenkins Dashboard

Workspace of pset-4-5 on master : / [Jenkins] - Mozilla Firefox

hoang-lenovo-ideapad-y700-touch-15isk:8080/job/pset-4-5/ws/

Jenkins

Back to Dashboard

Status

Changes

Workspace

Wipe Out Current Workspace

Build Now

Delete Project

Configure

GitHub

Rename

Build History trend

find

May 4, 2019, 8:51 PM

RSS for all RSS for failures

Workspace of pset-4-5 on master

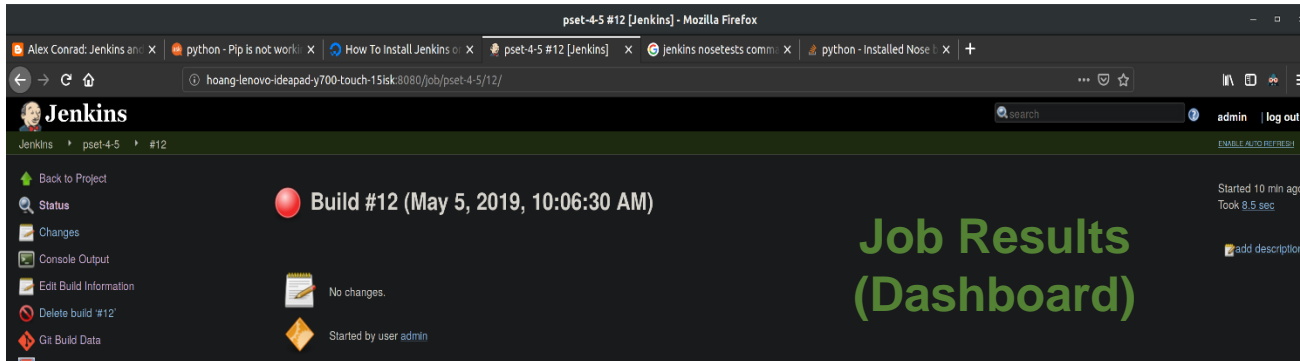
- git
- pyenv
- vscode
- cl
- docs
- reports
- src
- tests
- bumpversion.cfg May 4, 2019, 8:51:39 PM 503 B view
- cookiecutterrc May 4, 2019, 8:51:39 PM 2.11 KB view
- coveragerc May 4, 2019, 8:51:39 PM 174 B view
- editorconfig May 4, 2019, 8:51:39 PM 215 B view
- gitignore May 4, 2019, 8:51:39 PM 629 B view
- appveyor.yml May 4, 2019, 8:51:39 PM 2.60 KB view
- AUTHORS.rst May 4, 2019, 8:51:39 PM 64 B view
- CHANGELOG.rst May 4, 2019, 8:51:39 PM 86 B view
- CONTRIBUTING.rst May 4, 2019, 8:51:39 PM 2.37 KB view
- LICENSE May 4, 2019, 8:51:39 PM 1 B view
- MANIFEST.in May 4, 2019, 8:51:39 PM 348 B view
- Pipfile May 4, 2019, 8:51:39 PM 357 B view
- Pipfile.lock May 4, 2019, 8:51:39 PM 42.76 KB view
- README.md May 4, 2019, 8:51:39 PM 13.27 KB view
- README.rst May 4, 2019, 8:51:39 PM 1.96 KB view
- setup.cfg May 4, 2019, 8:51:39 PM 561 B view
- setup.py May 4, 2019, 8:51:39 PM 2.79 KB view
- tox.ini May 4, 2019, 8:51:39 PM 1.79 KB view

(all files in zip)

Local repository of codes ("cloned" from GitHub)

Page generated: May 4, 2019, 8:52:46 PM EDT Jenkins ver. 2.175

A07 “Integration” Jenkins Dashboard & Workspace



The Jenkins Dashboard for Build #12 (May 5, 2019, 10:06:30 AM) shows a successful build status. The left sidebar contains links to Back to Project, Status, Changes, Console Output, Edit Build Information, Delete build #12, Get Build Data, No Tags, Test Result, Violations, and Previous Build. The main area displays the build status as 'Completed', started 10 min ago, and took 8.5 sec. It also shows the revision: cc02f9f64bd4a6a3ba9783dc2cdefe30e8d70da and the test result: Test Result (1 failure) nose.failure.Failure.runTest.

Job Results (Dashboard)

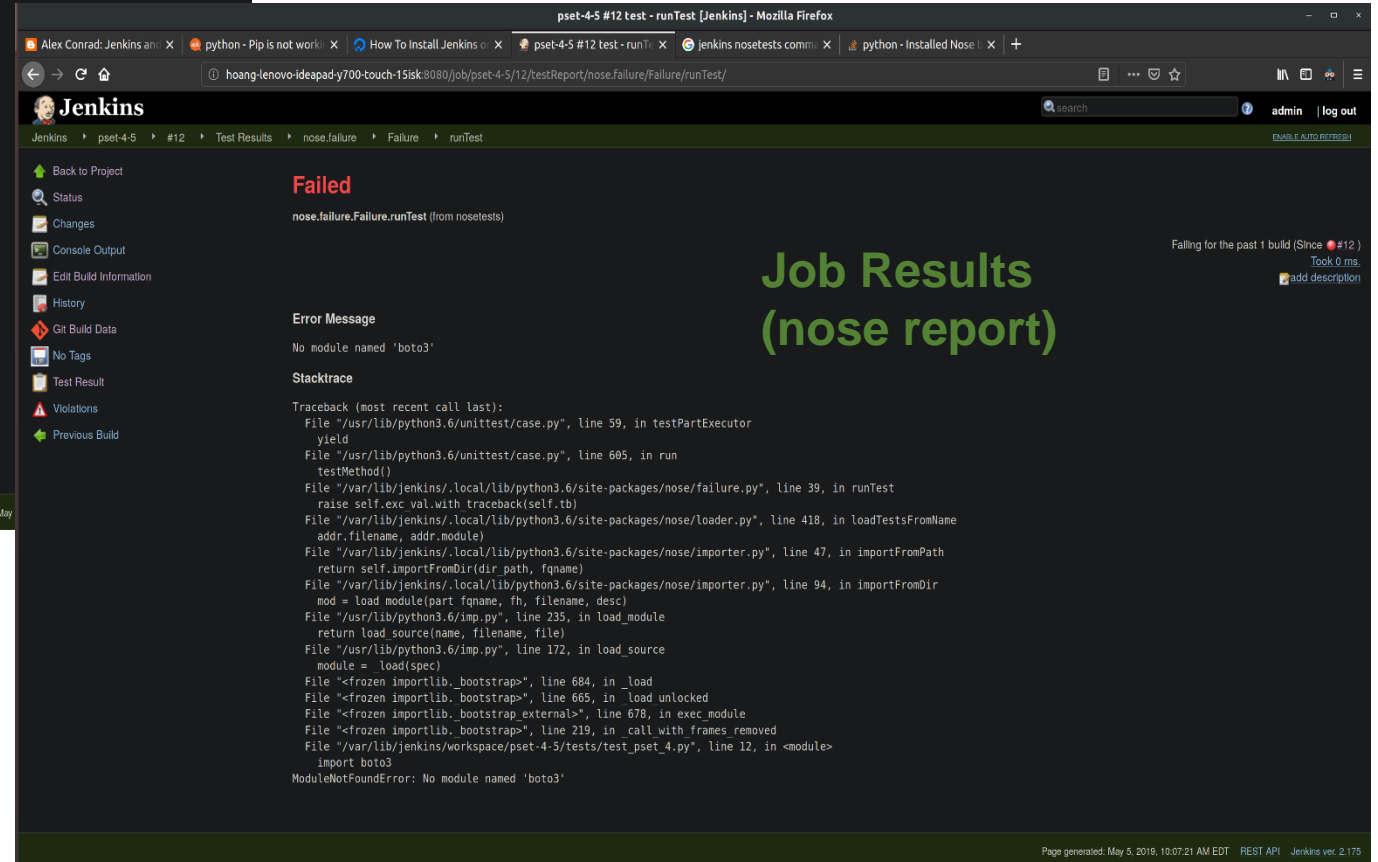
```
+ basename /var/lib/jenkins/workspace/pset-4-5/.pyenv
+ PS1=(.pyenv) $
+ export PS1
+ alias pydoc
+ [ -n ]
+ [ -n ]
+ pip3 install --quiet nosecoverage
+ pip3 install --quiet /var/lib/jenkins/workspace/pset-4-5/
+ /var/lib/jenkins/.local/bin/nosetests --with-xcoverage --with-xunit
E
=====
ERROR: Failure: ModuleNotFoundError (No module named 'boto3')
-----
Traceback (most recent call last):
  File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/failure.py", line 39, in runTest
    raise self.exc_val.with_traceback(self.tb)
  File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/loader.py", line 418, in loadTestsFromName
    addr.filename, addr.module)
  File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/importer.py", line 47, in importFromPath
    return self.importFromDir(dir_path, fqname)
  File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/importer.py", line 94, in importFromDir
    mod = load_module(part_fqname, fh, filename, desc)
  File "/usr/lib/python3.6/imp.py", line 235, in load_module
    return load_source(name, filename, file)
  File "/usr/lib/python3.6/imp.py", line 172, in load_source
    module = _load(spec)
  File "<frozen importlib._bootstrap>", line 684, in _load
  File "<frozen importlib._bootstrap>", line 665, in _load_unlocked
  File "<frozen importlib._bootstrap_external>", line 678, in exec_module
  File "<frozen importlib._bootstrap>", line 219, in _call_with_frames_removed
  File "/var/lib/jenkins/workspace/pset-4-5/tests/test_pset_4.py", line 12, in <module>
    import boto3
ModuleNotFoundError: No module named 'boto3'
```

Name	Stats	Miss	Branch	BrPart	Cover	Missing
neural_style/___init___py	0	0	0	0	100.00%	
pset_4/___init___py	1	0	0	0	100.00%	
pset_4/tasks/___init___py	1	0	0	0	100.00%	
TOTAL	2	0	0	0	100.00%	

```
Ran 1 test in 0.023s

FAILED (errors=1)
Build step 'Execute shell' marked build as failure
[Cobertura] Publishing Cobertura coverage report...
No reports were found
[htmlpublisher] Archiving HTML reports...
[htmlpublisher] Archiving at PROJECT level /var/lib/jenkins/workspace/pset-4-5/reports to /var/lib/jenkins/job
Recording test results
Finished: FAILURE
```

Job Results (terminal report)



The Jenkins Test Results page for Build #12 shows a failed test result. The left sidebar contains links to Back to Project, Status, Changes, Console Output, Edit Build Information, History, Git Build Data, No Tags, Test Result, Violations, and Previous Build. The main area displays the test result as 'Failed' with the error message: 'No module named 'boto3''. The stacktrace shows the error occurred in the testPsetExecutor, yield, testMethod(), and in the runTest method of the failure.py module.

Failed

nose.failure.Failure.runTest (from nosetests)

Failing for the past 1 build (Since #12)
Took 0 ms

Job Results (nose report)

Error Message

No module named 'boto3'

Stacktrace

Traceback (most recent call last):
File "/usr/lib/python3.6/unittest/case.py", line 59, in testPartExecutor
yield
File "/usr/lib/python3.6/unittest/case.py", line 605, in run
testMethod()
File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/failure.py", line 39, in runTest
raise self.exc_val.with_traceback(self.tb)
File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/loader.py", line 418, in loadTestsFromName
addr.filename, addr.module)
File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/importer.py", line 47, in importFromPath
return self.importFromDir(dir_path, fqname)
File "/var/lib/jenkins/.local/lib/python3.6/site-packages/nose/importer.py", line 94, in importFromDir
mod = load_module(part_fqname, fh, filename, desc)
File "/usr/lib/python3.6/imp.py", line 235, in load_module
return load_source(name, filename, file)
File "/usr/lib/python3.6/imp.py", line 172, in load_source
module = _load(spec)
File "<frozen importlib._bootstrap>", line 684, in _load
File "<frozen importlib._bootstrap>", line 665, in _load_unlocked
File "<frozen importlib._bootstrap_external>", line 678, in exec_module
File "<frozen importlib._bootstrap>", line 219, in _call_with_frames_removed
File "/var/lib/jenkins/workspace/pset-4-5/tests/test_pset_4.py", line 12, in <module>
import boto3
ModuleNotFoundError: No module named 'boto3'

Page generated: May 5, 2019, 10:07:21 AM EDT REST API Jenkins ver. 2.175

Congratulations

